Novel anomaly detection and classification algorithms for IP and mobile networks

Agathe Blaise

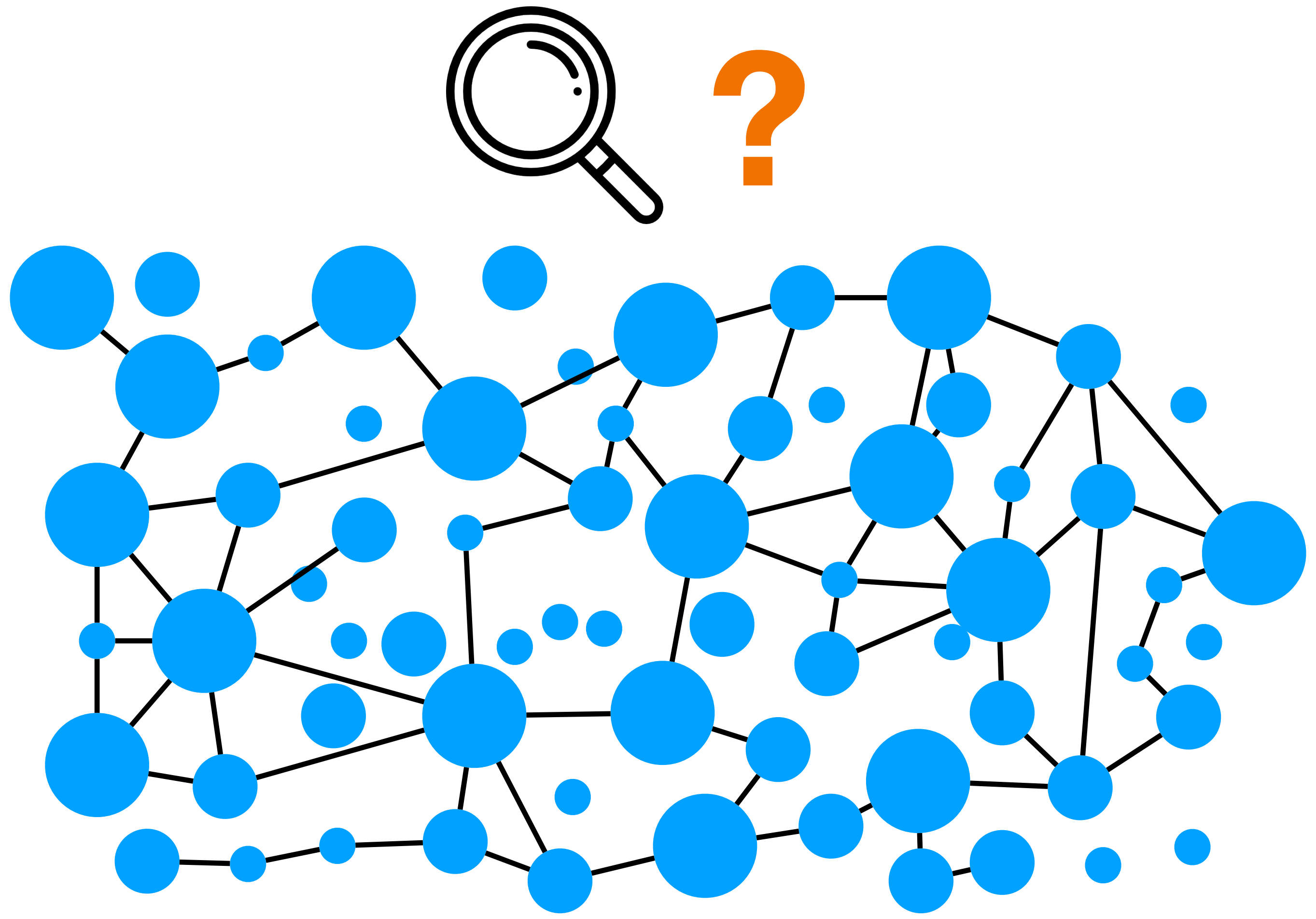
Thesis defended on December 14, 2020 before a jury composed of:

Marco Fiore, IMDEA Networks	Reviewer
Razvan Stanica, INSA Lyon, Inria	Reviewer
Clémence Magnien, CNRS, Sorbonne Université	Examiner
Aline Carneiro Viana, Inria Saclay	Examiner
Sahar Hoteit, Université Paris Saclay, Centrale-Supélec	Examiner
Sandra-Scott Hayward, Queen University Belfast	Invited member
Thi-Mai-Trang Nguyen, CNRS, Sorbonne Université	Invited member
Stefano Secci, Conservatoire National des Arts et Métiers	Supervisor
Vania Conan, Thales	Supervisor
Mathieu Bouet, Thales	Supervisor

Data analysis

Data: logs of communications, list of transactions, actions of the users, etc.

Potentially **thousands of logs** to handle each day

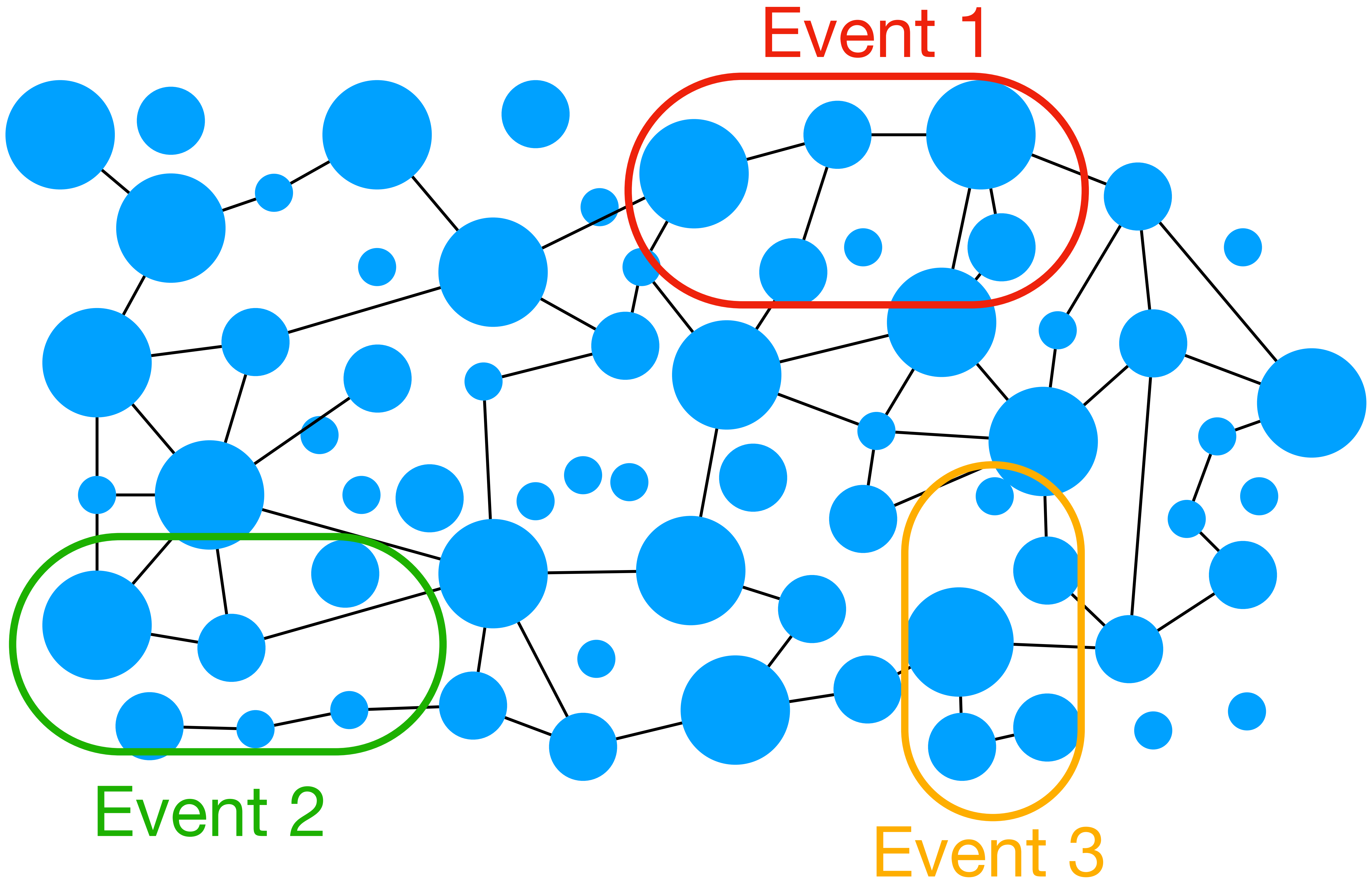


At first sight: **indecipherable**
and no obvious patterns

Knowledge discovery:

- Find underlying patterns
- Define generic model for learning

Data analysis techniques



Numerous anomalies

- Correlate them to find **events**
- Investigate **root causes**, **identity** of attackers, modus operandi…

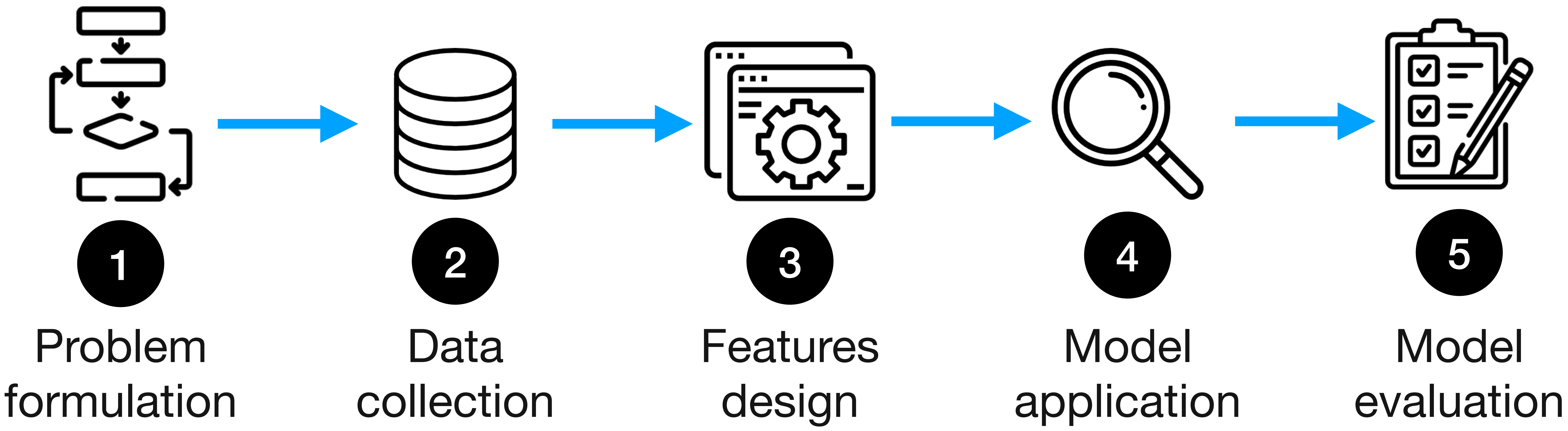
State-of-the-art: rather complex, fine-grained approaches
e.g., neural networks, graph-based techniques

⚠ Very expensive computationally and not fit for real networks

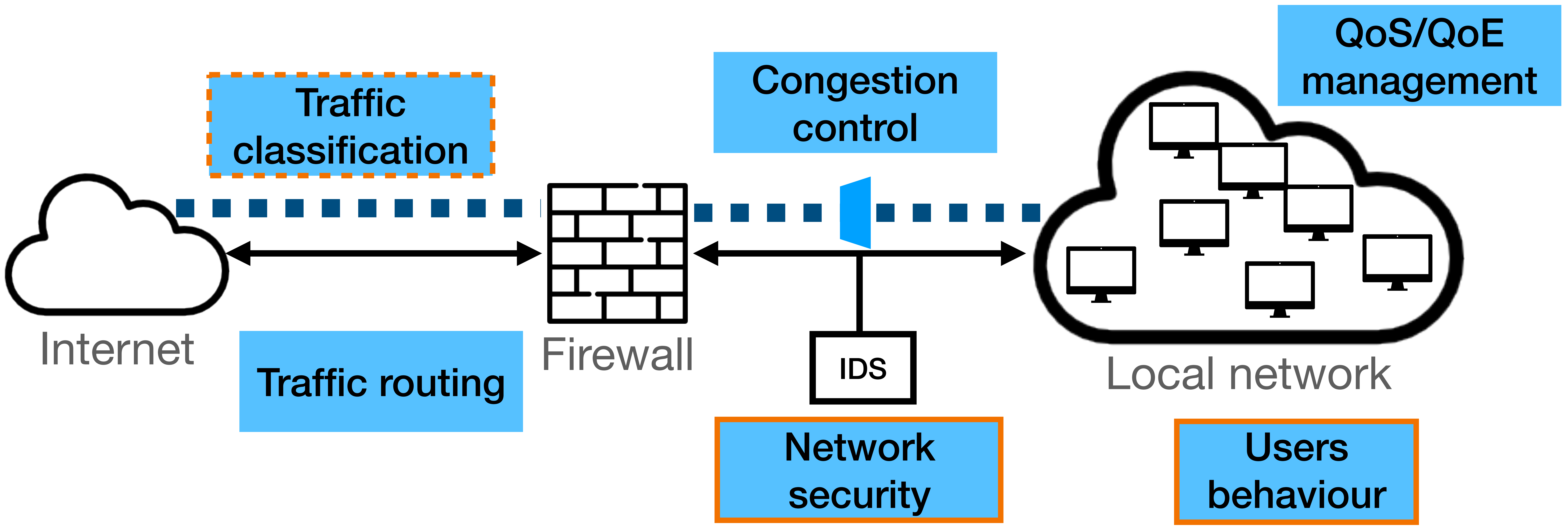
Data analysis techniques

- **Statistical** techniques
- **Machine Learning** techniques

"A computer can be programmed so that it will learn to play a better game of checkers than can be played by the person who wrote the program.” - Arthur Samuel (1959)



Network behaviour analysis



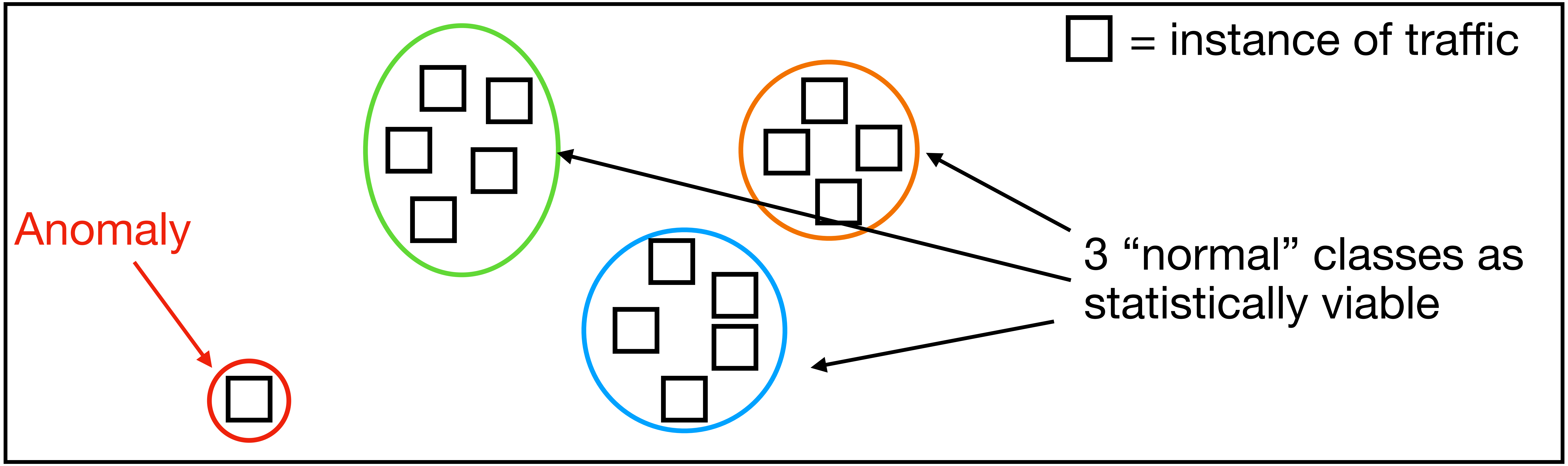
Targets of data analysis

- **Malicious behaviour** from users
  - Denial-of-service attacks, network scanning

- **Unusual behaviour** from users
  - Bursts of traffic, special events, point-to-multipoint communications

- **Operational events**
  - Outages from the network or cloud operator, hardware failures, bad configurations

Data analysis



1. Aggregation level

□ = host, flow?

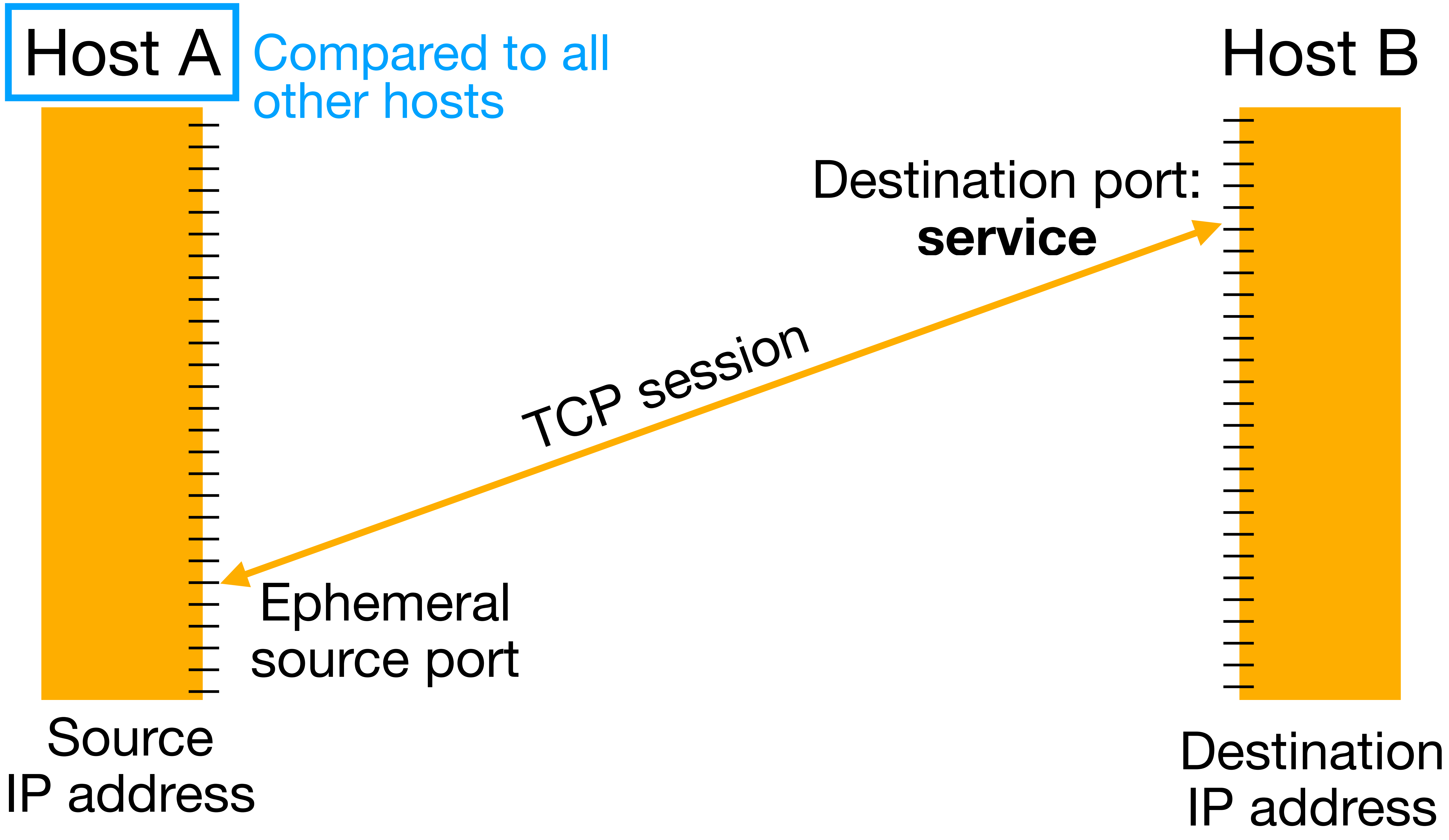
What to characterise?

2. Features choice

→ Attributes of the element

How to characterise it?

Aggregation levels



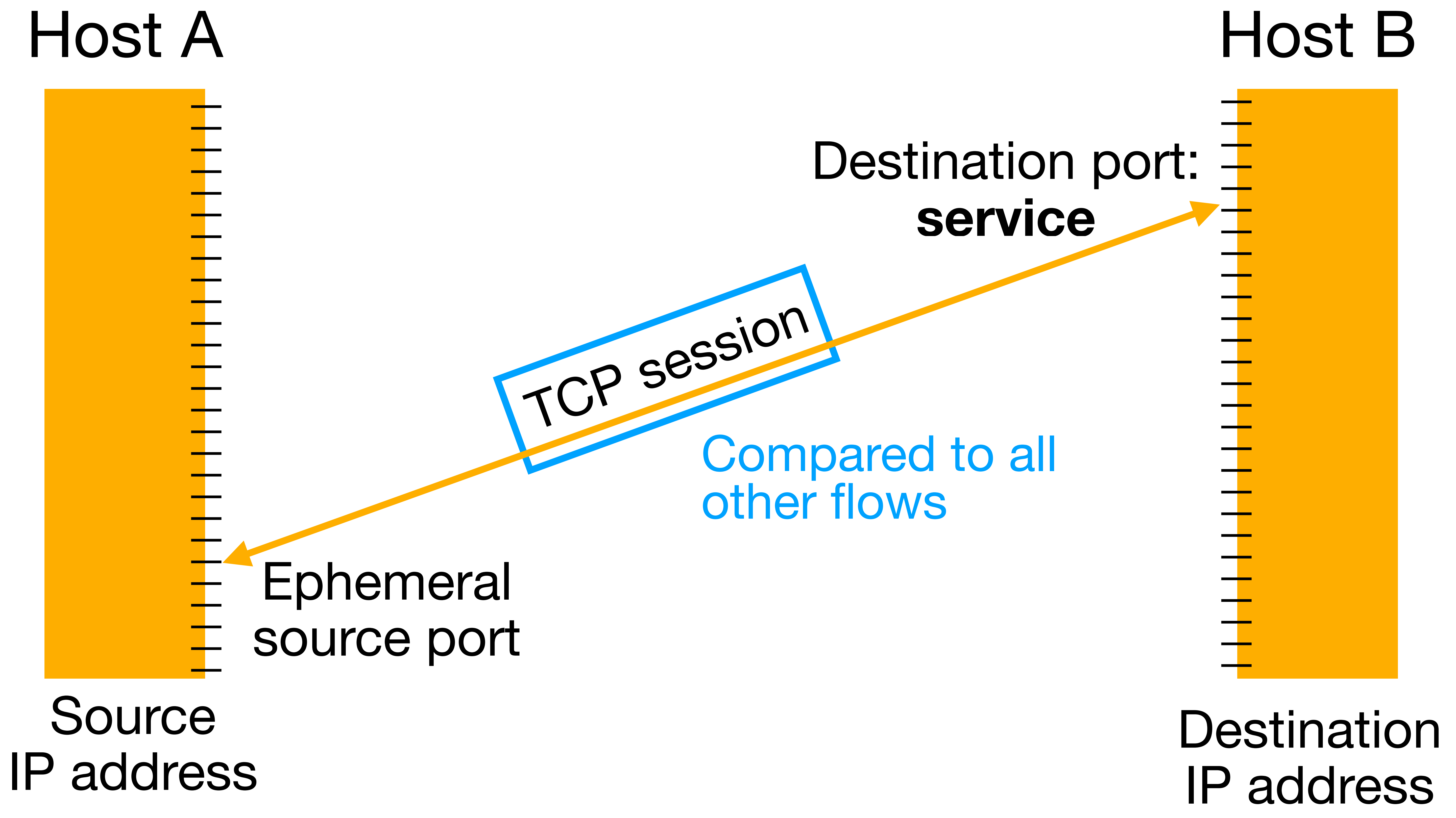
1. **Aggregation level**

Host behaviour

2. **Features**

Packet counts, frequency of communications, protocols

Aggregation levels



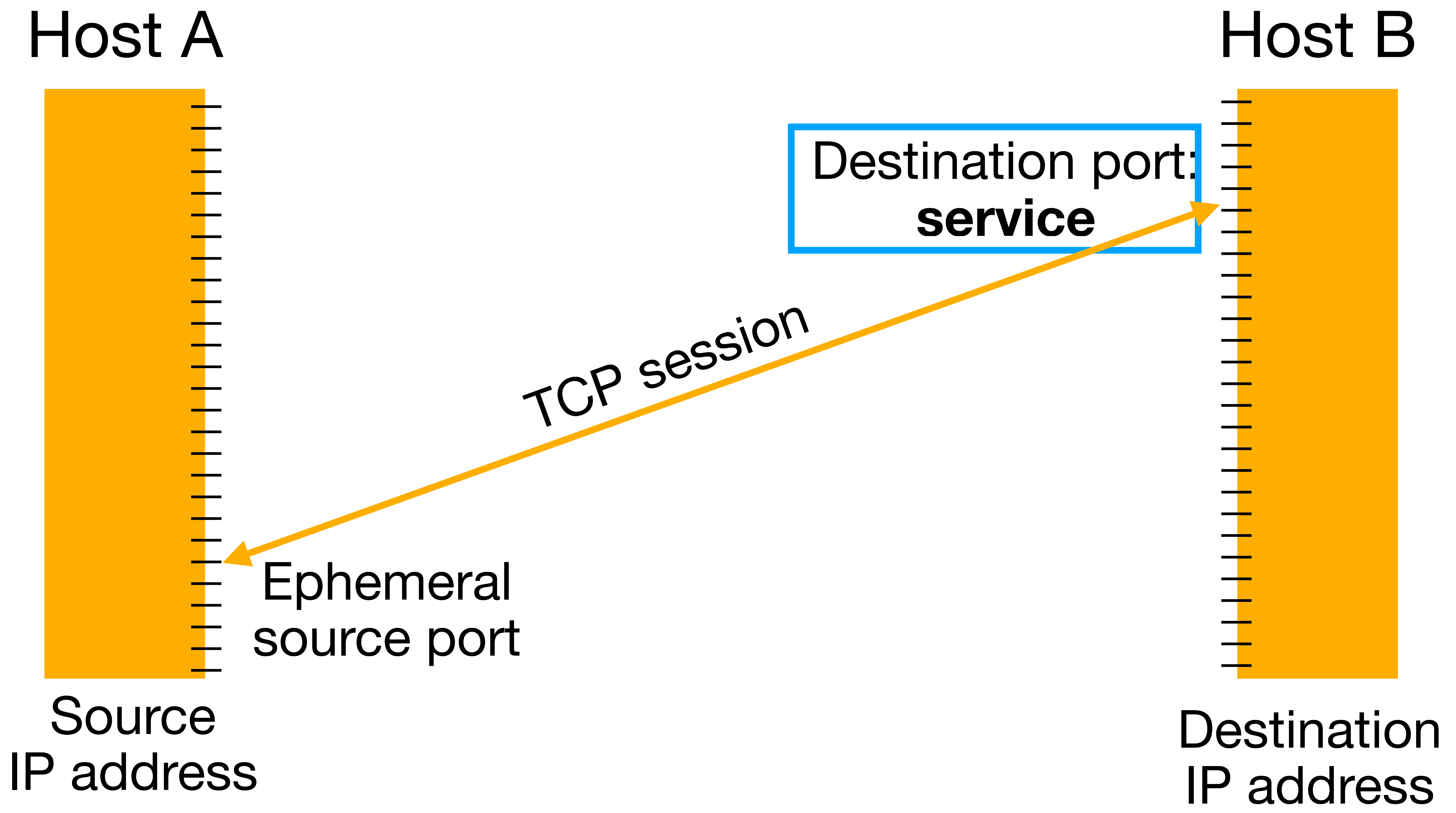
1. **Aggregation level**

Flow features

2. **Features**

Flow duration, flow volume, mean packet length, packet inter-arrival time, entropy

Aggregation levels



1. **Aggregation level**

2. **Features**

→ Port or service-level **rarely analyzed**

Contributions outline

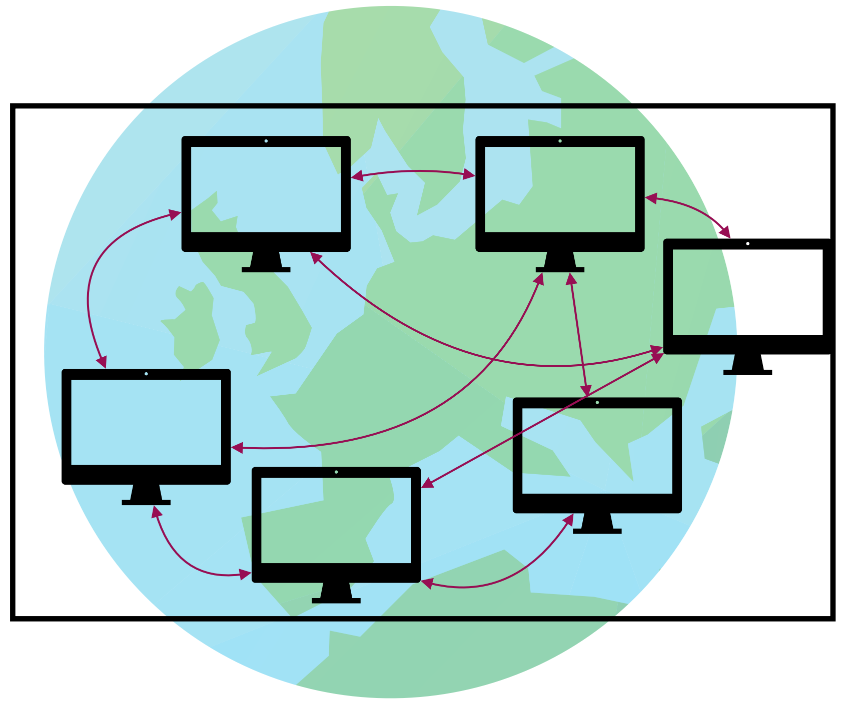
Analysis of the usage of **services, applications and port numbers**

- **State-of the art:** reasons why unused technique
- **Our objective:** assessing its **benefits** through **lightweight** techniques
- Our contributions in 3 different contexts:

Split-and-Merge

Internet-carrier level

BotFingerPrinting



Local (corporate) network

ASTECH

Cellular networks

Security aspects

Behavioural analysis

Per-service detection

Rather **underused** method:

- Numerous elements to analyse
  - In IP networks: 65,536 ports
  - In cellular networks: all services or mobile apps

→ Requires an algorithm of low-complexity

- Traffic obfuscation to avoid firewalls

→ Concerns only a few cases

- Encrypted traffic

→ Deep Packet Inspection to induce used applications

Per-service detection

Ports and applications **universally and permanently** used

Able to identify uncommon behaviours **not seen with flows and IP adresses**:

- **Long-term** detection as ports subsist over time

→ Detection of attackers **slowly spreading**

- **Several vantage points** as ports universally used

→ **Cross-validation**

- Application **failover** or **update**, **vulnerability scan** on a given port

→ Not visible by analysing IP addresses and flows

Our objectives

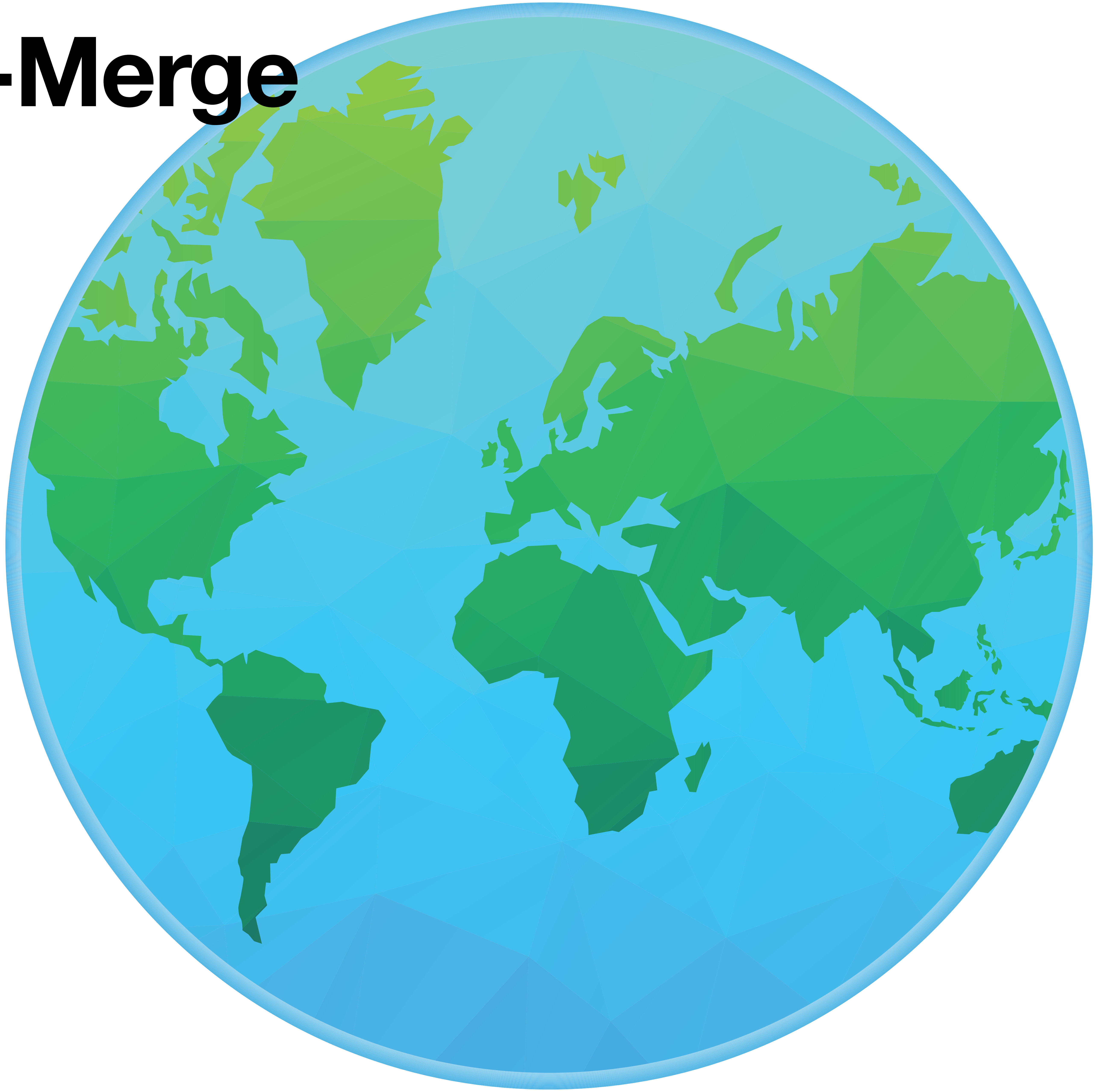
- **State-of-the art:** complex approaches, not fit for real networks

Objective: provide a **pragmatic approach**, lightweight, efficient and scalable

- Through the analysis of **ports, services and applications** usages
- Using **statistical and machine learning** techniques: classification, clustering, anomaly detection
- In various contexts: at IP-level, in local networks, in cellular networks

Split-and-Merge

Split-and-Merge



Split-and-Merge

At Internet carrier-level

- Detection of large-scale **attacks**: vulnerability scans
- Trend of major **botnets** spreading

Split-and-Merge

<u>**Challenge**</u>: major botnets spreading **not detected** by traditional Intrusion Detection Systems

<u>**Our approach:**</u>

- **Long-term** analysis of ports usage
- **Cross-validation** in several subnetworks

<u>**Our contribution:**</u> detection of large-scale **vulnerability scans** and **botnets** spreading

Server vulnerabilities

Exposed to the Internet, open ports, no authentication

Common Vulnerabilities and Exposures:

- CVE-2018-1000115 (memcached) port 11211
- CVE-2017-17215 (Huawei HG532 routers) port 37215

IoT devices vulnerabilities

Low computational power to run **security functions**

- CVE-2018-7445 (MikroTik devices) port 8291
- CVE-2018-11653 & CVE-2018-11654 (Netwave IP cameras) port 8000

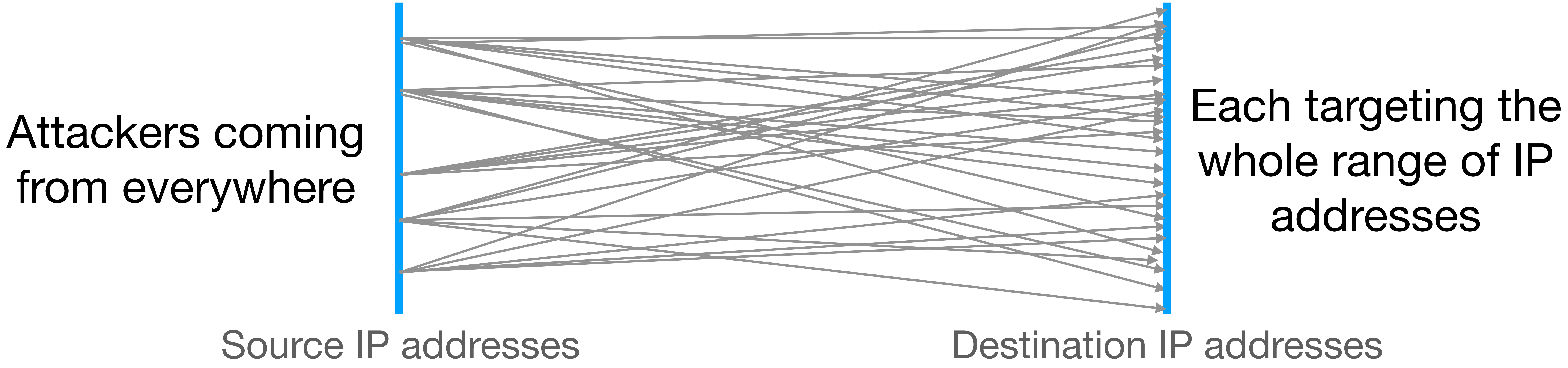
→ Identification of these services or devices by port number.

Context

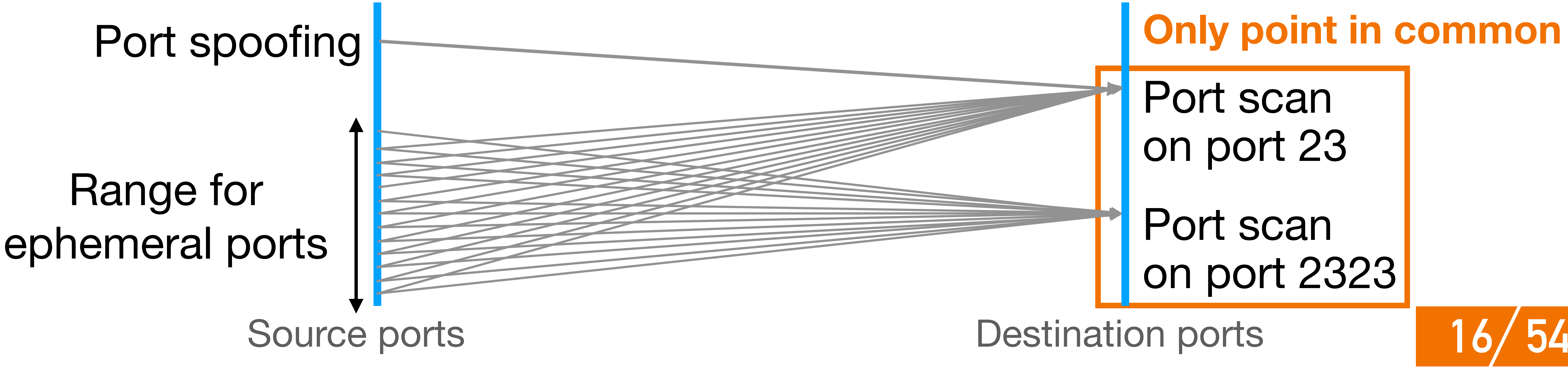
Vulnerability scan

Port scan to identify devices hosting **vulnerable services**

- IP addresses



- Port numbers



Split-and-Merge

Overview

- **Long-term analysis of the usage of ports:**

  1 - Features computation

  2 - Local anomaly detection

  3 - Central correlation

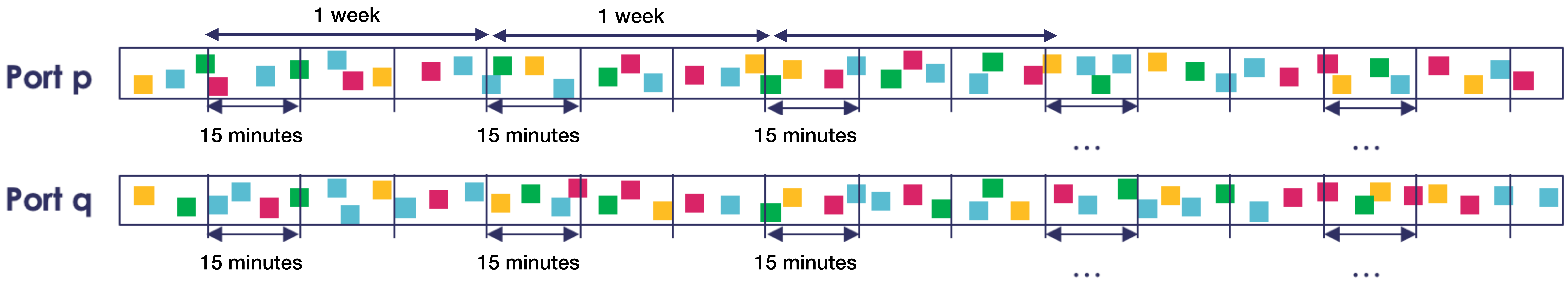
  4 - Fine-grained anomaly characterisation

Split-and-Merge

1 - Features computation

For each port *p:*

- Source diversity index
- Destination diversity index
- Port diversity index
- Mean packet size
- Standard deviation of packet size
- Percentage of SYN packets

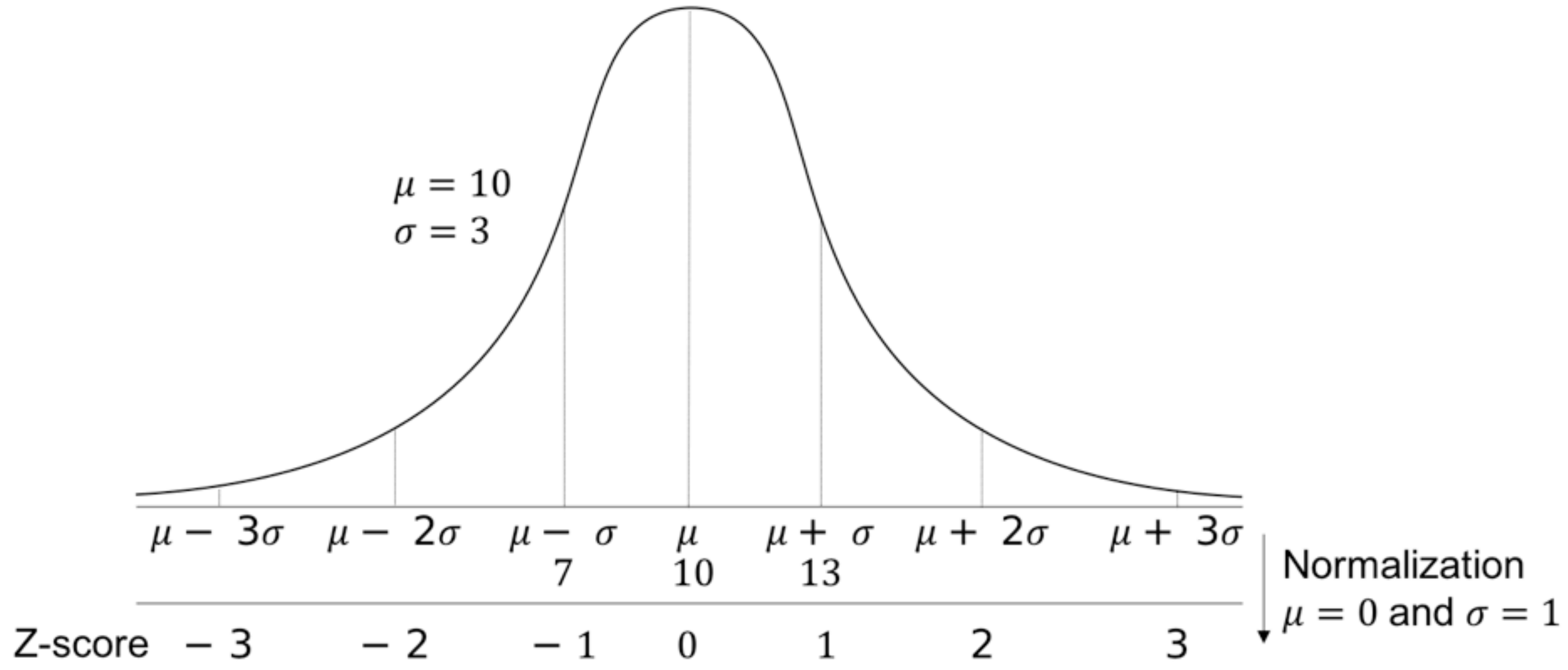


Split-and-Merge

2 - Local anomaly detection

Time series x → normal distribution $\mathcal{N}(\mu, \sigma^2)$ of mean μ and std σ

port *p*	x_1	x_2	x_3
Feature	7	13	30
Feature	54	50	53



- Z-score of x_i : $Z = \dfrac{x_i - \mu}{\sigma}$

  → **not resistant to outliers**

- Modified Z-score using median and median std

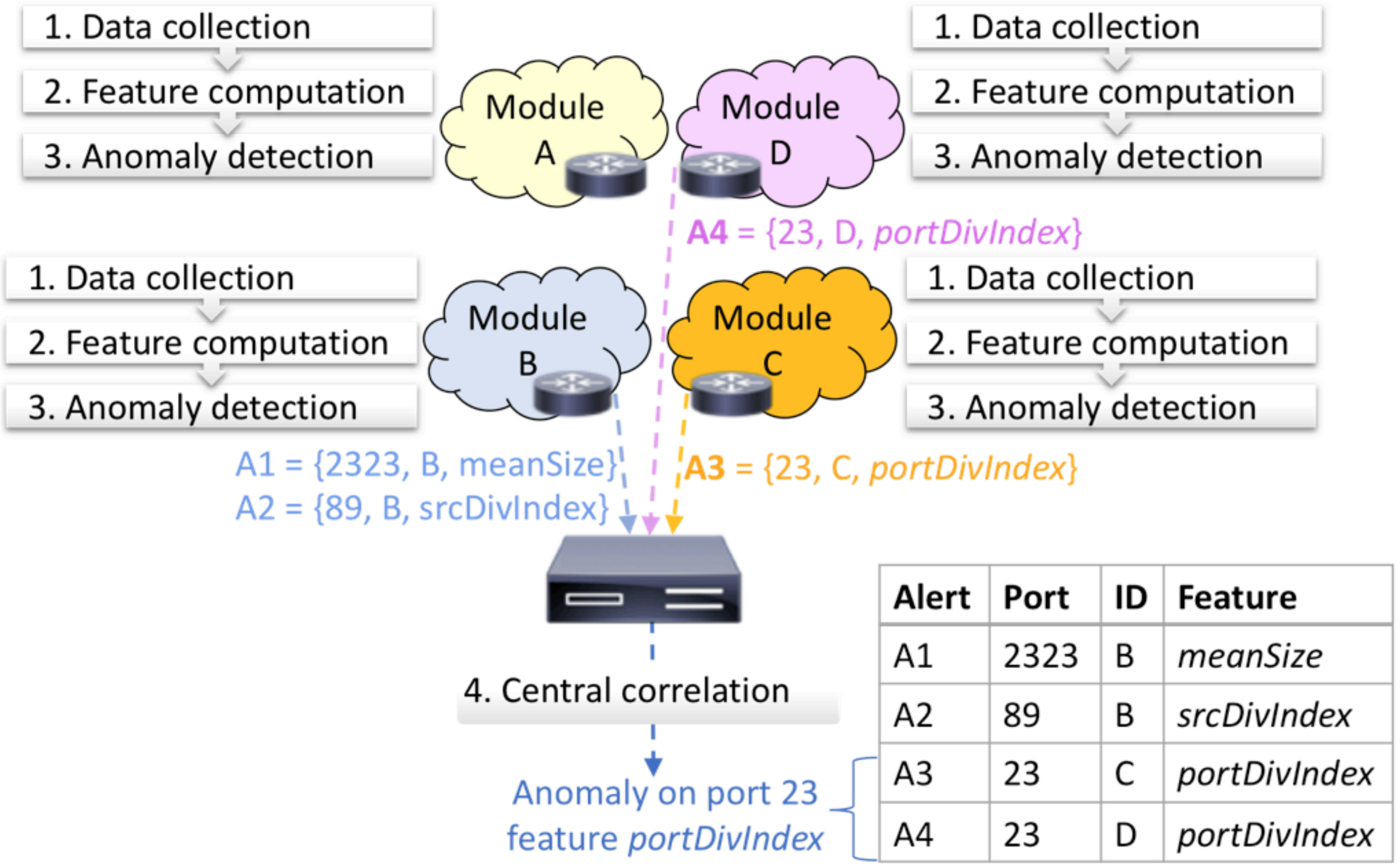
If M > threshold T = 3.5 → **anomaly**

Split-and-Merge

3 - Central correlation

To reduce false positives: Split-and-Merge architecture

Central controller: keep only **distributed** anomalies



Alert	Port	ID	Feature
A1	2323	B	*meanSize*
A2	89	B	*srcDivIndex*
A3	23	C	*portDivIndex*
A4	23	D	*portDivIndex*

Split-and-Merge

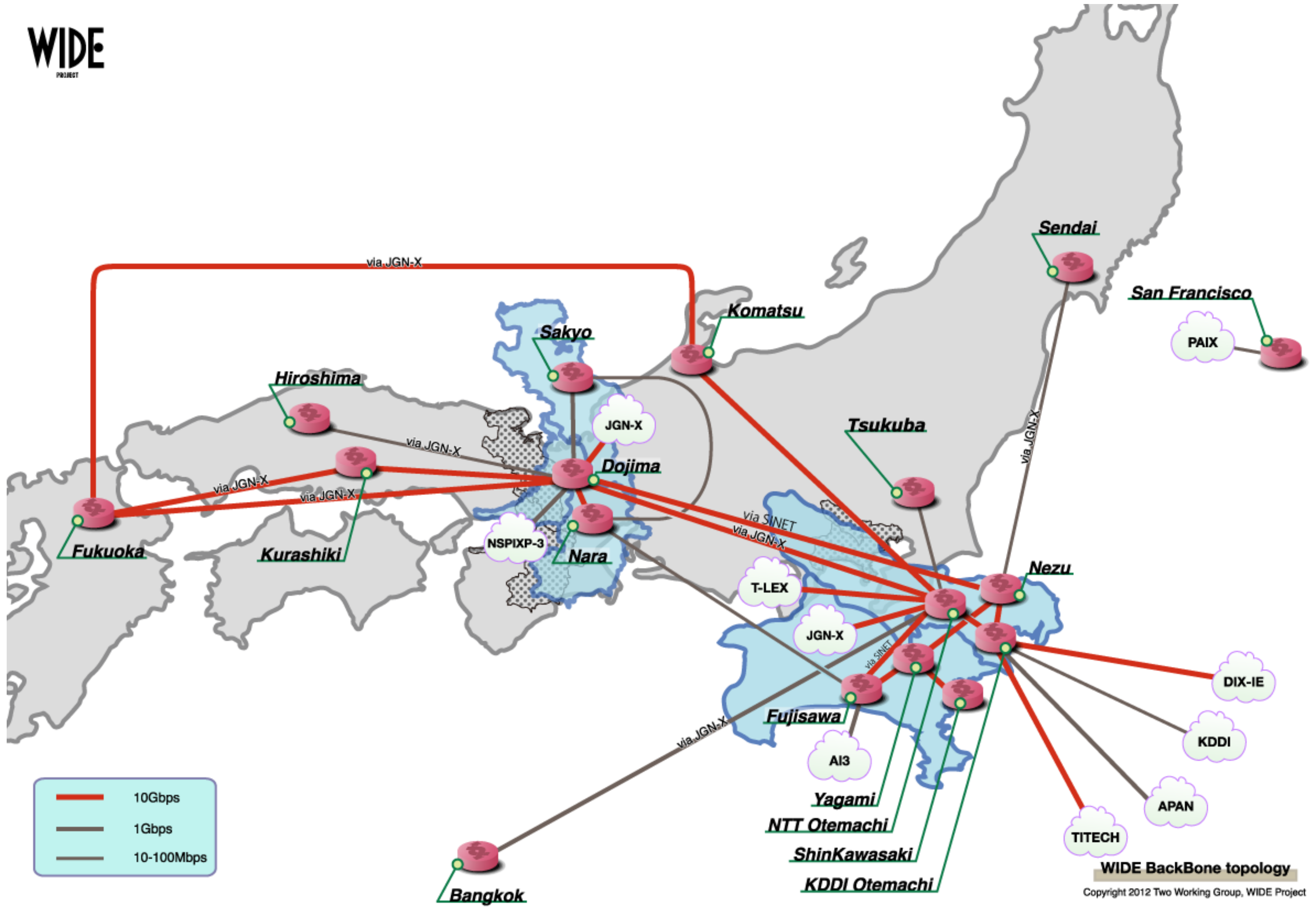
4 - Fine-grained characterisation through expert rules

Classes	Characteristics
More normal packets	`+meanSize, +stdSize`
More forged packets	`-meanSize, -stdSize`
Large scan	`-srcDivIndex, +destDivIndex, -meanSize`
DDoS	`+srcDivIndex, -destDivIndex`
Botnet scan	`+srcDivIndex, +destDivIndex, -meanSize`
Botnet expansion	`+srcDivIndex, +destDivIndex, -stdSize`
Targeted scan	`-srcDivIndex, -destDivIndex`
Less botnet scan	`-srcDivIndex, -destDivIndex, +meanSize, -stdSize`

Evaluation on real-world traces

MAWI dataset (WIDE Project):

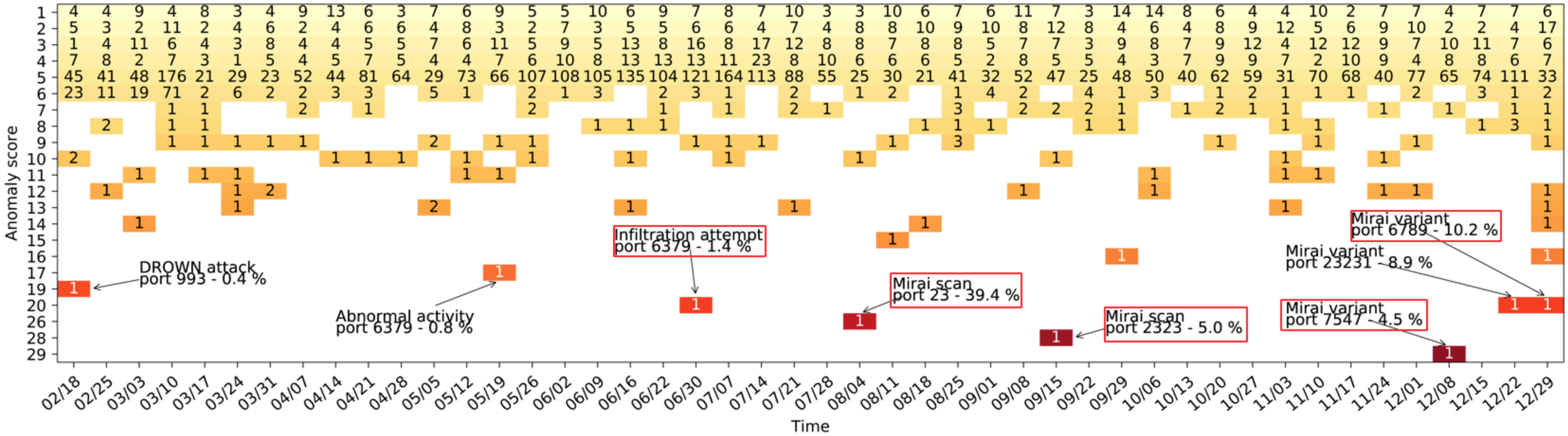
- **Daily files** of 15 minutes of traffic from a transpacific link
- Captured between the **MAWI network and the upstream ISP**



Evaluation (2016)

Anomaly score: number of anomalies for one port

→ Considering **all subnetworks** and **all features**



- Very low number of anomalies
- **Not detected** by traditional IDSs (MAWILab, ORUNADA)

MAWILab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking, *Co-NEXT*, 2010.

Online and Scalable Unsupervised Network Anomaly Detection Method, *IEEE Transactions on Networks and Service Management*, 2016.

Retrospective of major botnets

- Mirai (ports 23, 2323, 7547, 6789, 2222, 23231)
- Hajime (port 5358)
- Reaper (port 20480)
- Satori (ports 37215, 52869)
- ADB.Miner (port 5555)
- Memcached (port 11211)
- Satori (port 8000)

Split-and-Merge conclusion

Benefits of **per-port detection:**

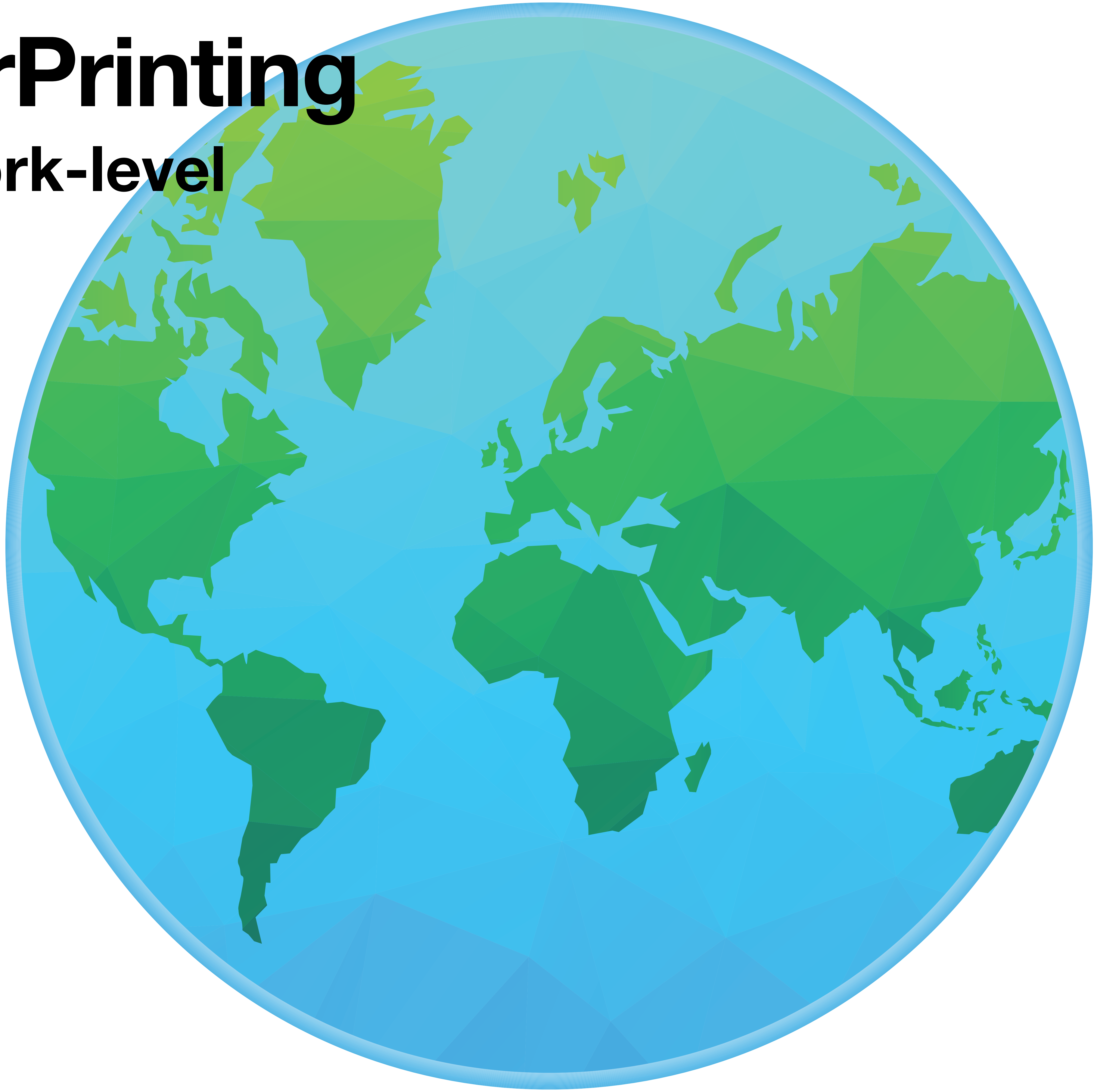
- Focus on **port numbers:** detection of **world-wide attacks**, not seen by traditional IDS
- **Long-term** analysis: possible only when using **port numbers**
- **Cross-validation** in different subnetworks: very few **false positives**

Lightweight algorithm: ideally running at the switch-level

BotFingerPrinting

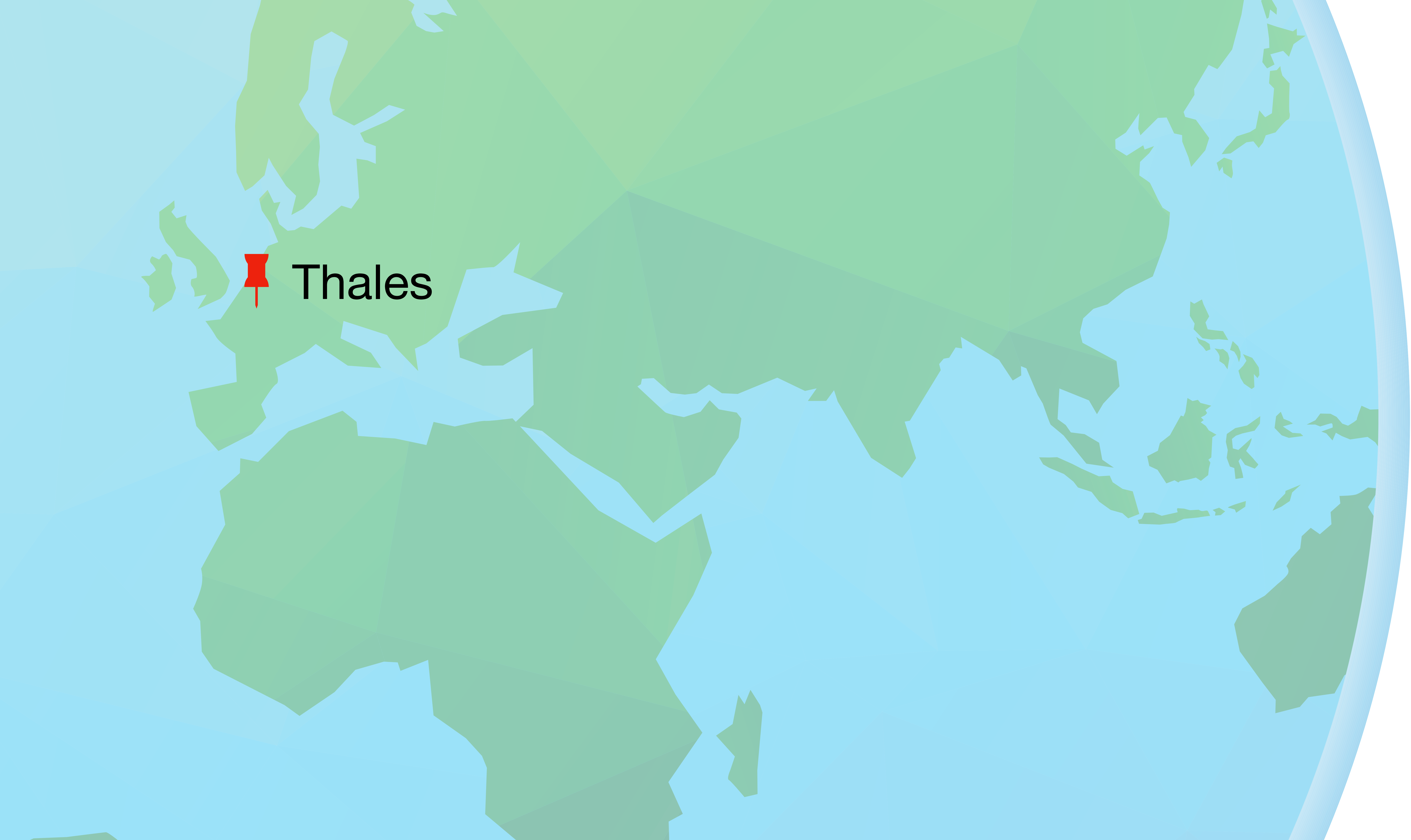
BotFingerPrinting

At local network-level



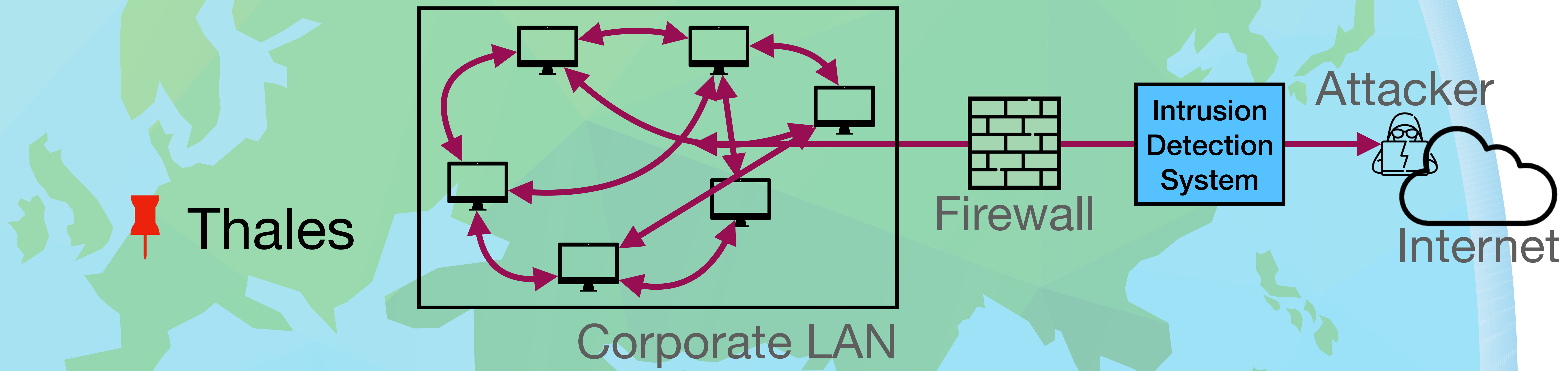
BotFingerPrinting

At local network-level



BotFingerPrinting

At local network-level



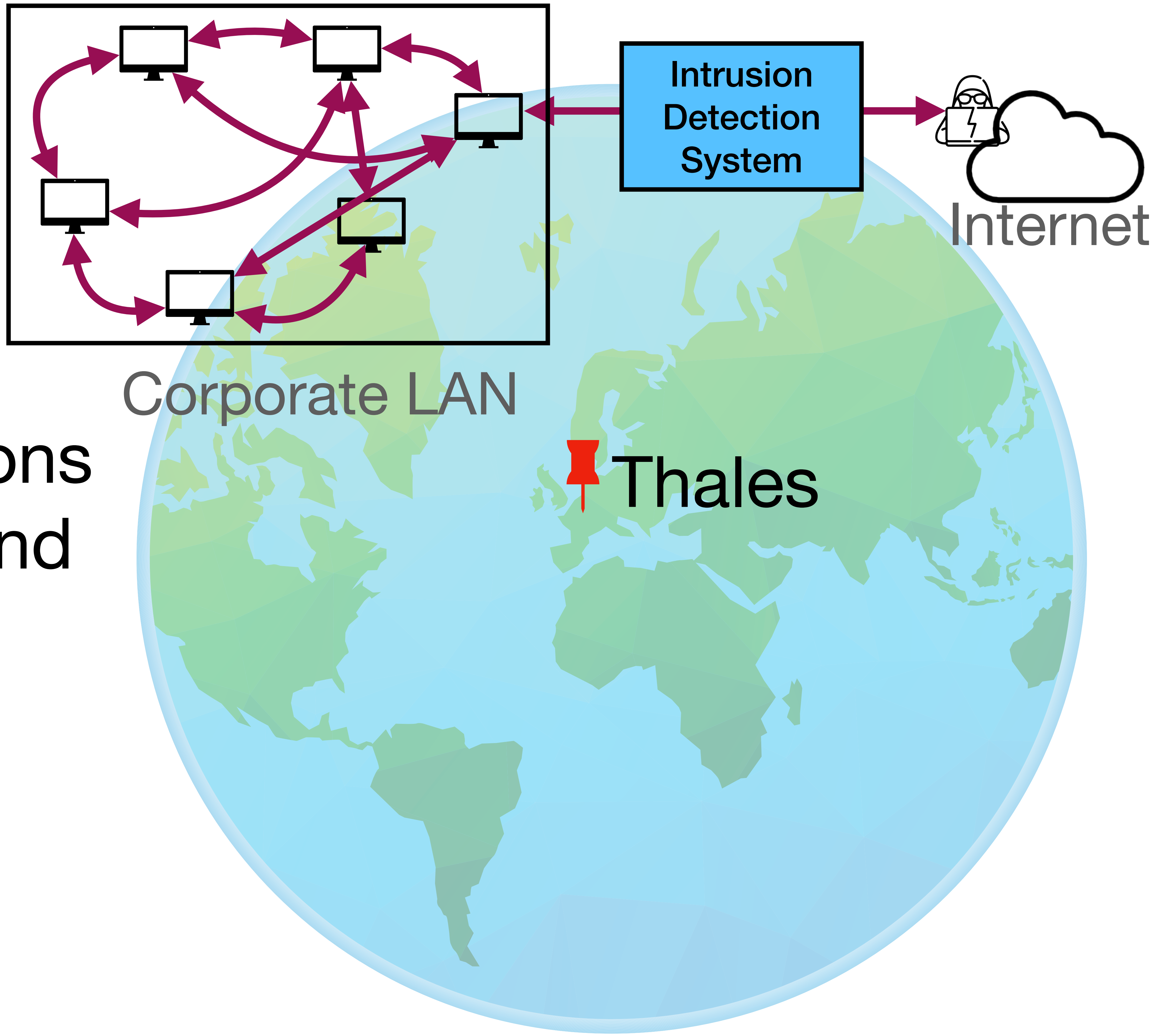
- **Model the communications** within a network
- Suspicious communications patterns to **find infected hosts**

BotFingerPrinting

<u>Challenge</u>: botnet detection within LAN

- <u>Flow-based</u> approaches: miss communications patterns
- <u>Graph-based</u> approaches: not scaling

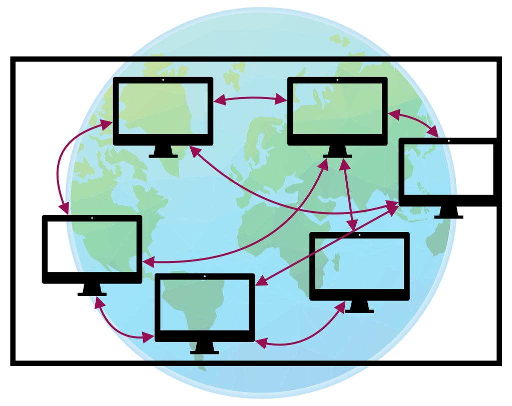
<u>Our approach:</u> simplify the communications graphs through histograms about hosts and **services contacted**



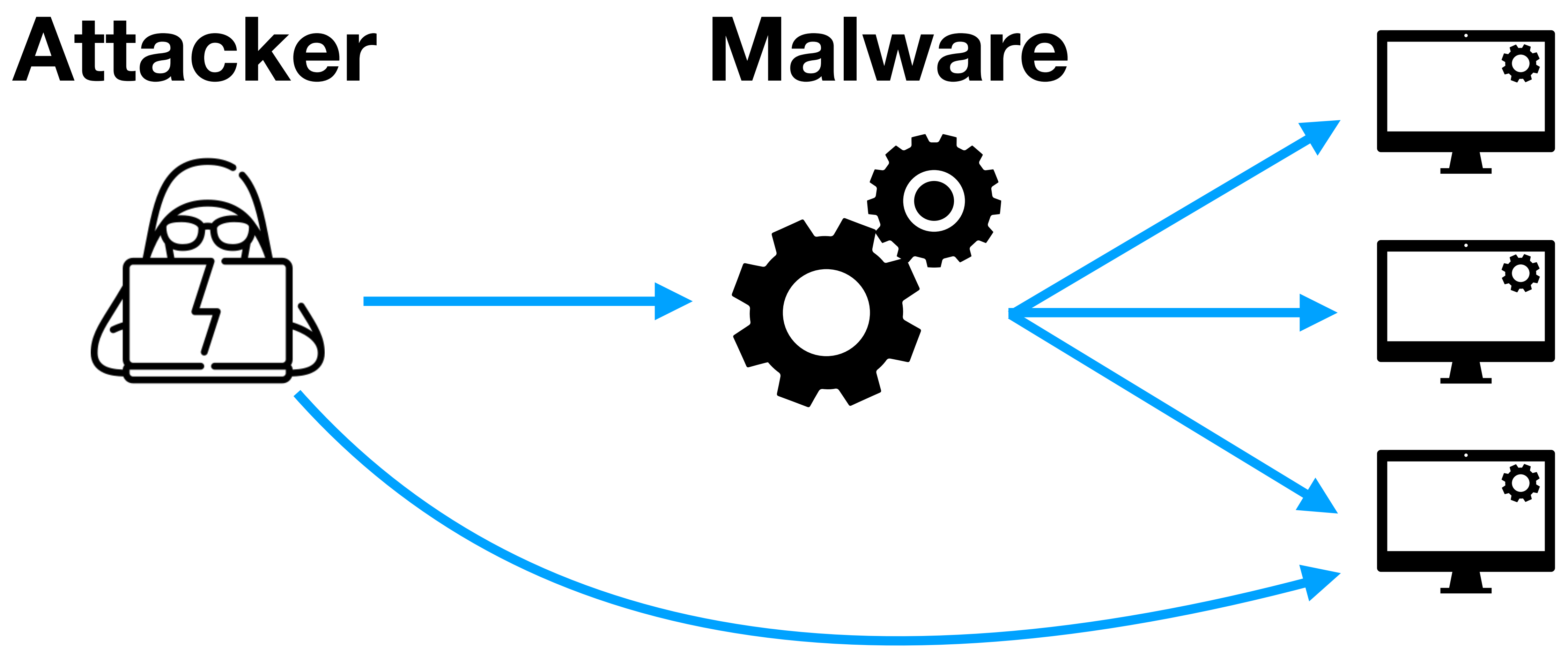
<u>Our contributions:</u>

- Very high accuracy compared to SOTA
- Lightweight compared to graph-based approaches

Context



Botnet architecture



Infected hosts

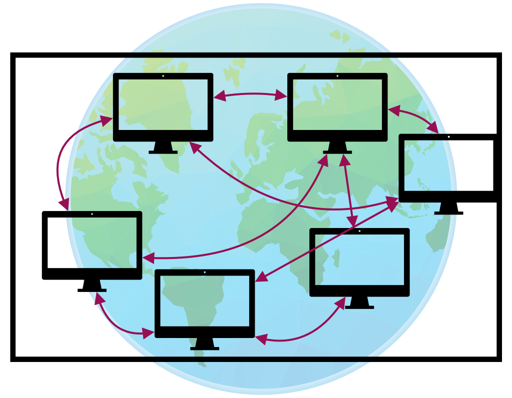
- **Malicious activities:** DDoS, spam, scan
- **Infection** of other hosts

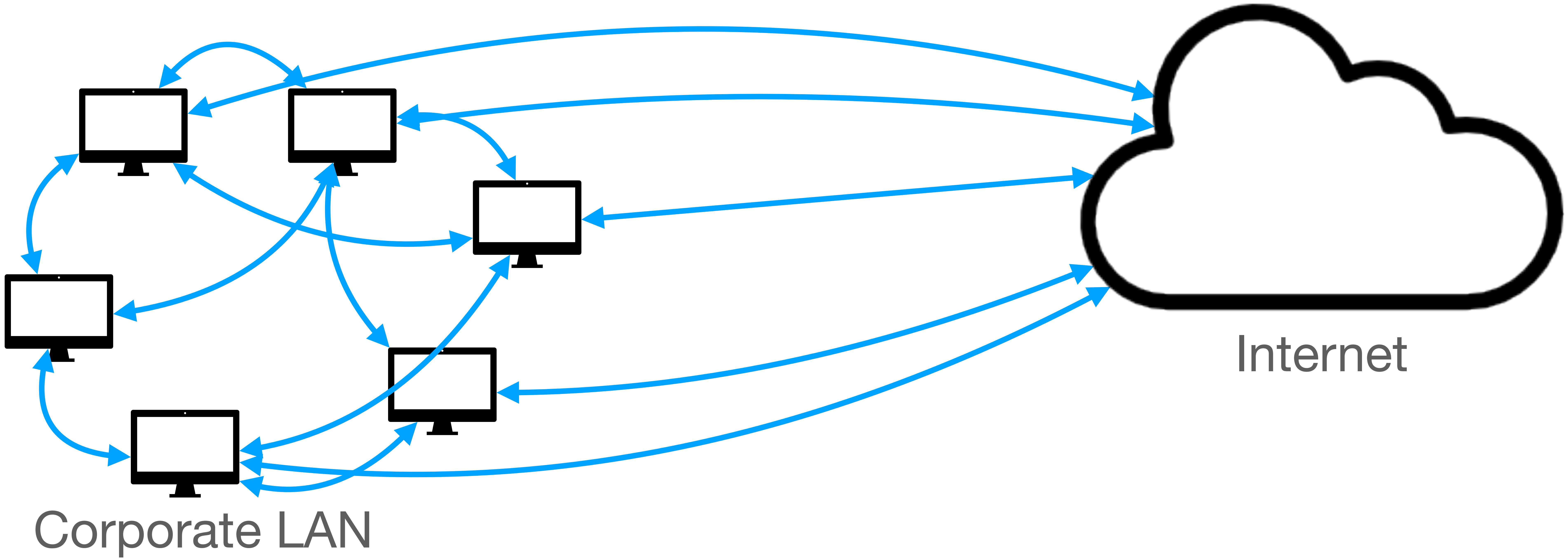
Command-and-Control (**C&C**) channel

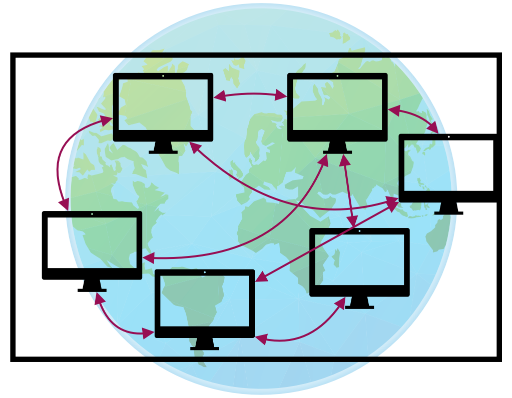
→ Need to identify **communication patterns specific to a bot.**

Graph-based approaches

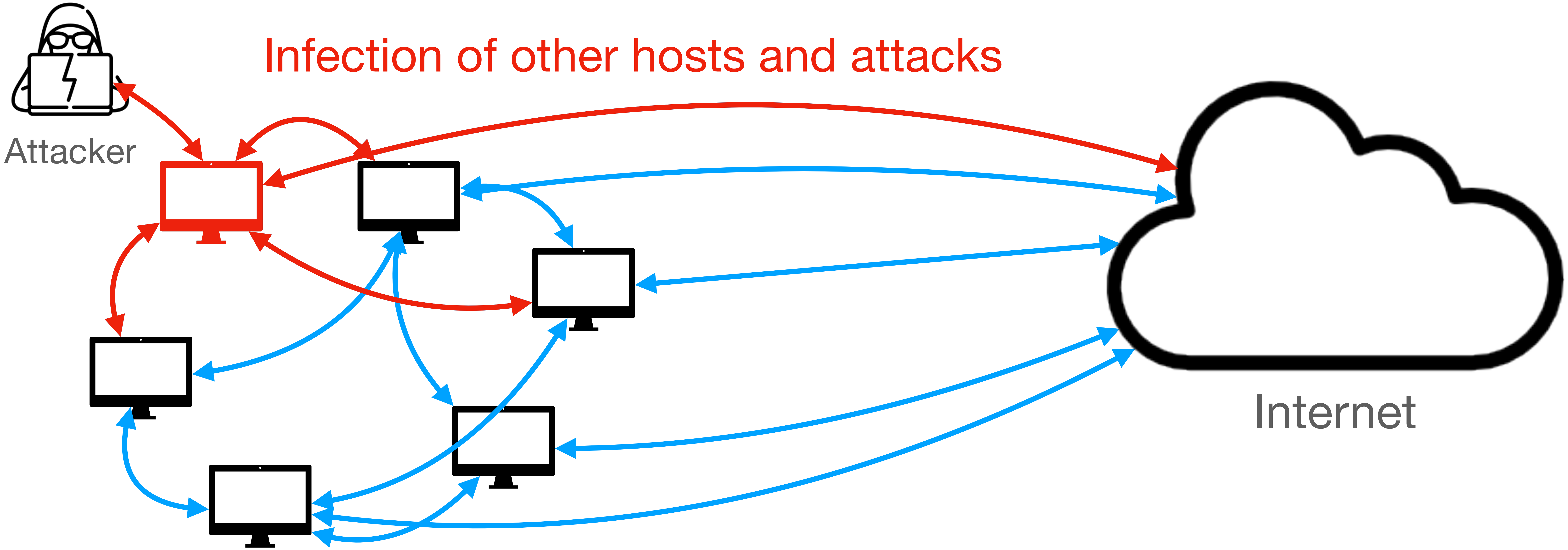
Context







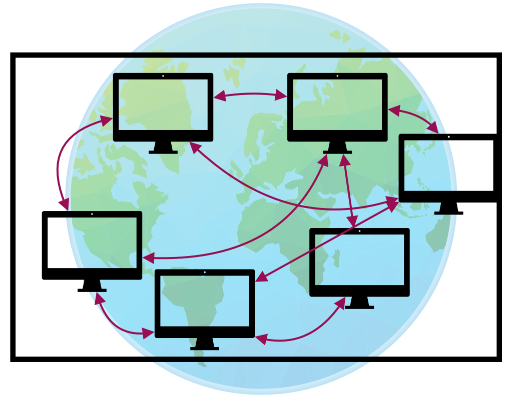
Graph-based approaches



- **Graphs** modelling the communications of an host
- Abnormal graphs ⇔ **botnets** → ***NP*-complete or cubic complexity**

→ Our objective: simplifying the communications graphs

Dataset



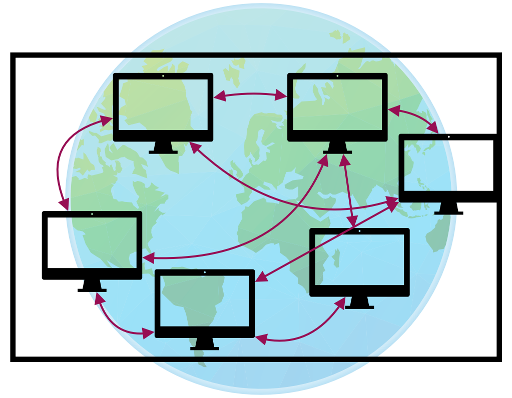
CTU-13 dataset (2011)

13 botnet scenarios: training and test (*) sets

Id	#bots	Malware	Activity
1*	1	Neris	IRC, SPAM, CF
2*	1	Neris	IRC, SPAM, CF
3	1	Rbot	IRC, PS
4	1	Rbot	IRC, DDoS
5	1	Virut	SPAM, PS
6*	1	Menti	PS
7	1	Sogou	HTTP
8*	1	Murlo	PS
9*	10	Neris	IRC, SPAM, CF, PS
10	10	Rbot	IRC, DDoS
11	3	Rbot	IRC, DDoS
12	3	NSIS.ay	IRC, P2P
13	1	Virut	HTTP, SPAM, PS

- **C&C channels:** IRC, HTTP, P2P
- **Malicious activities:** DDoS, port scan, spam, click fraud

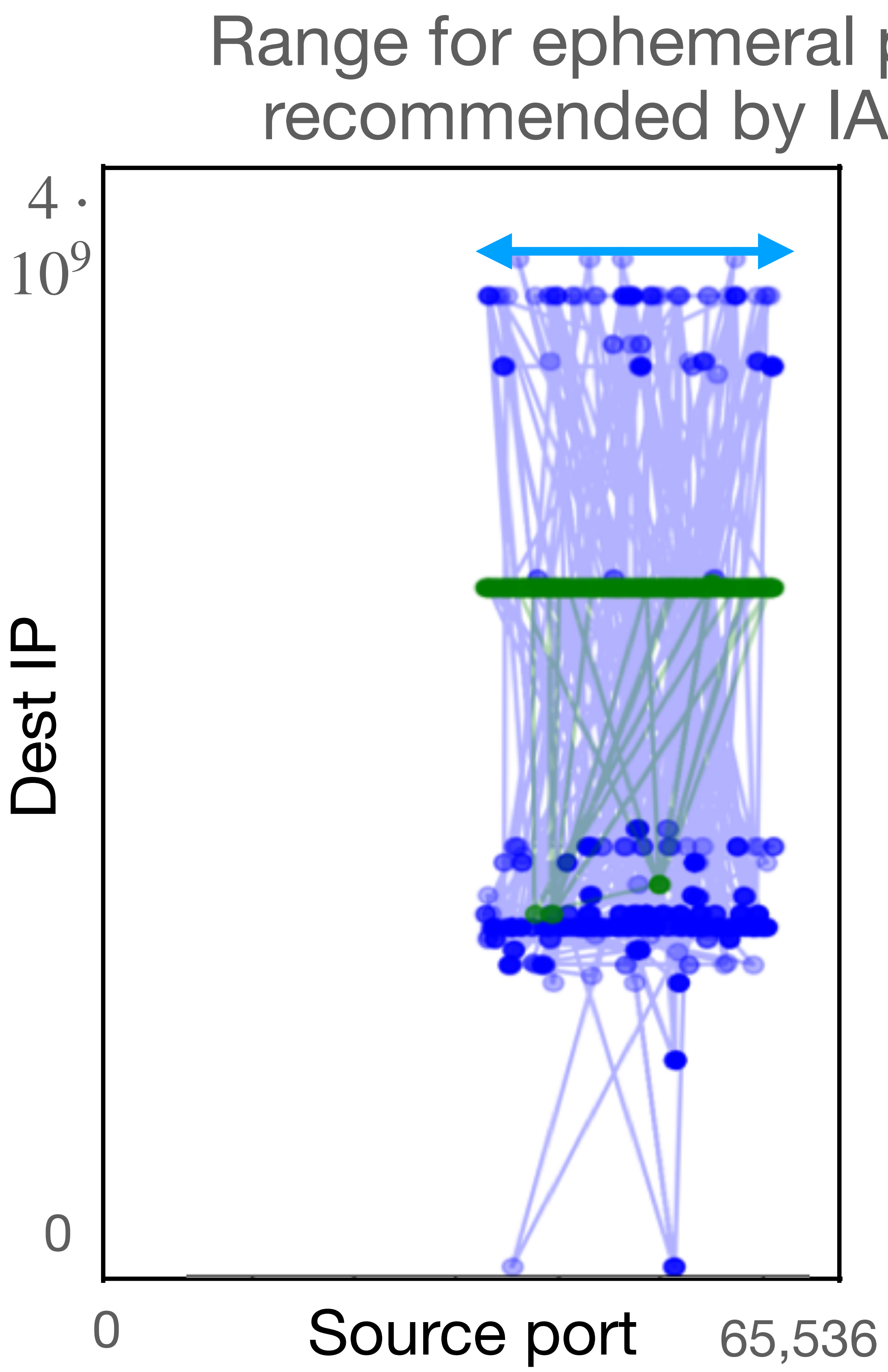
→ **Objective:** learn from training set and perform the detection on test set.



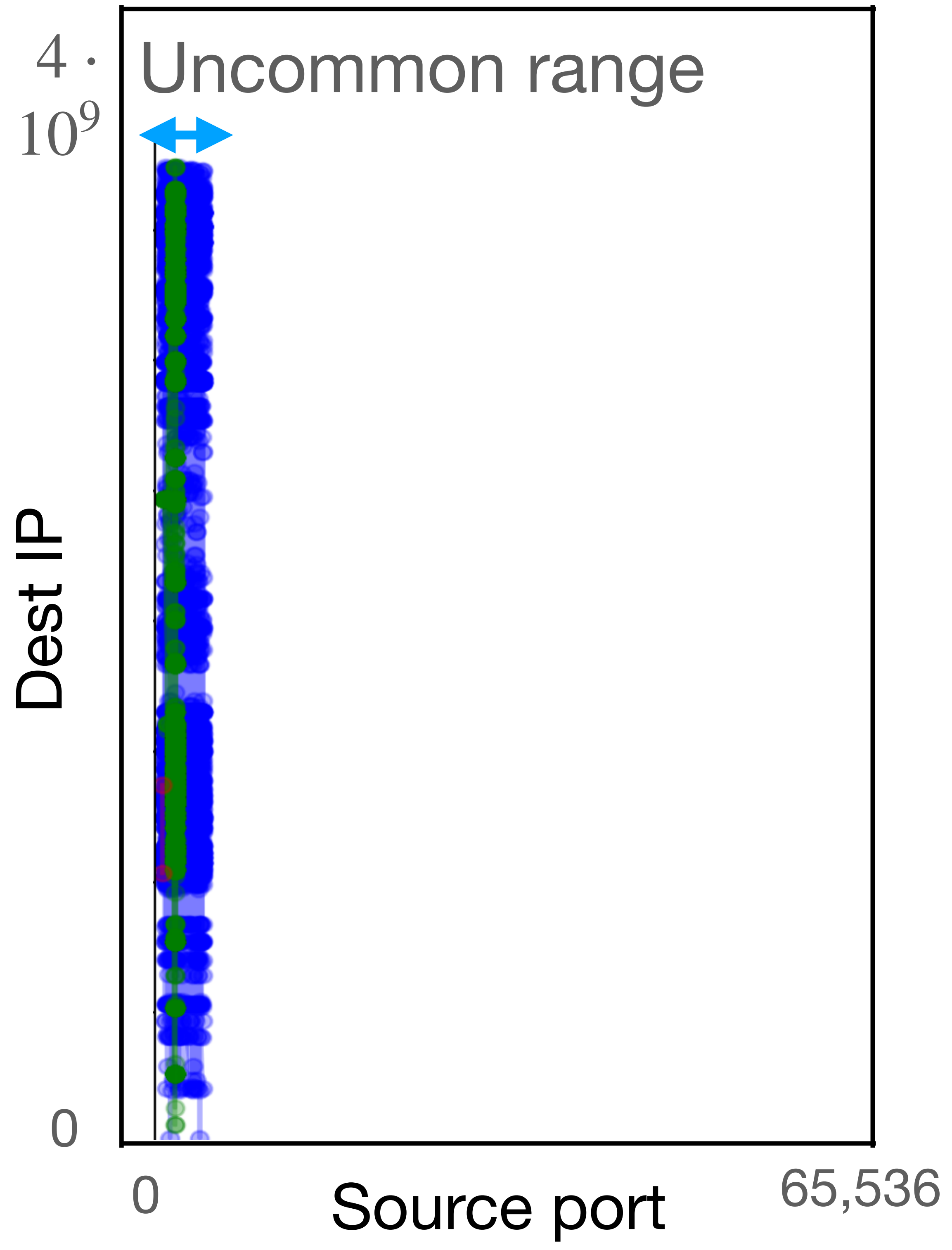
First observations on CTU-13

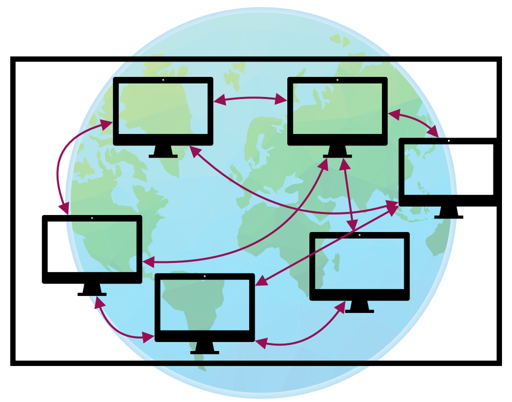
Inspecting the communications of two different hosts

Benign host



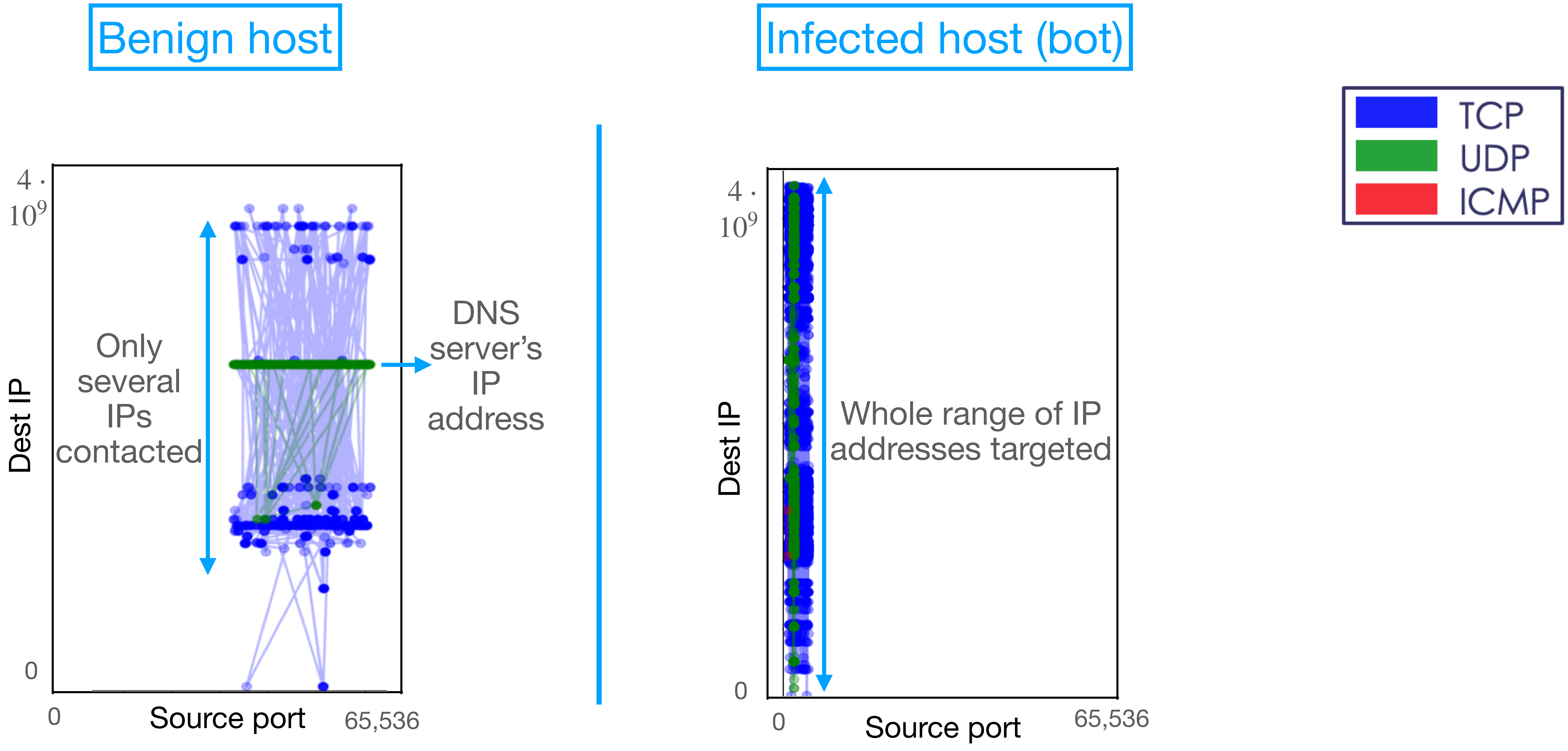
Infected host (bot)

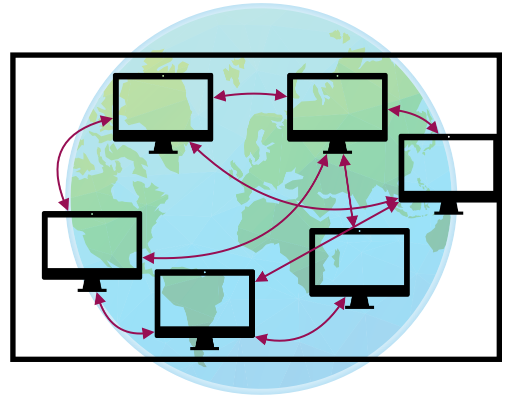




First observations on CTU-13

Inspecting the communications of two different hosts

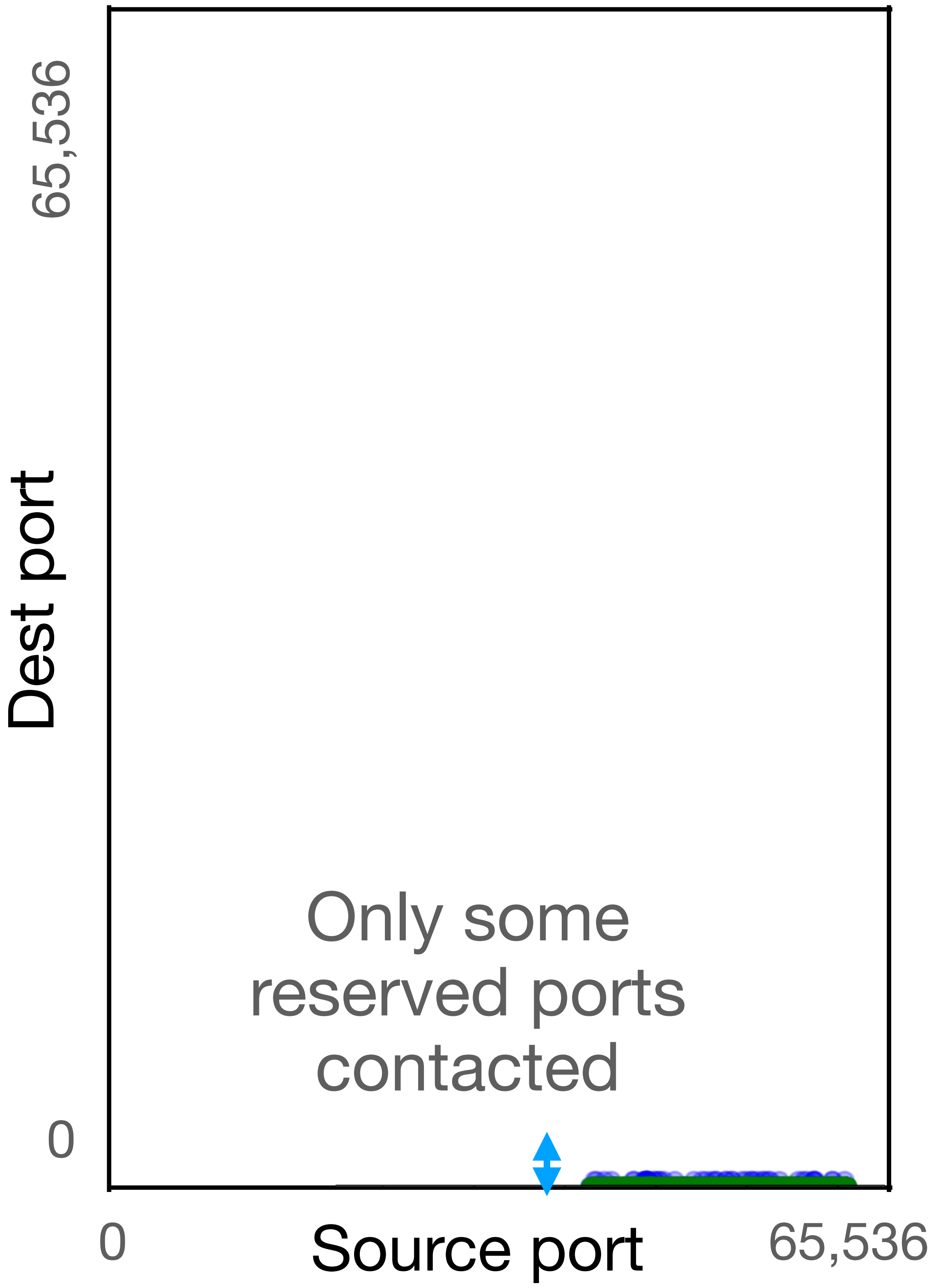




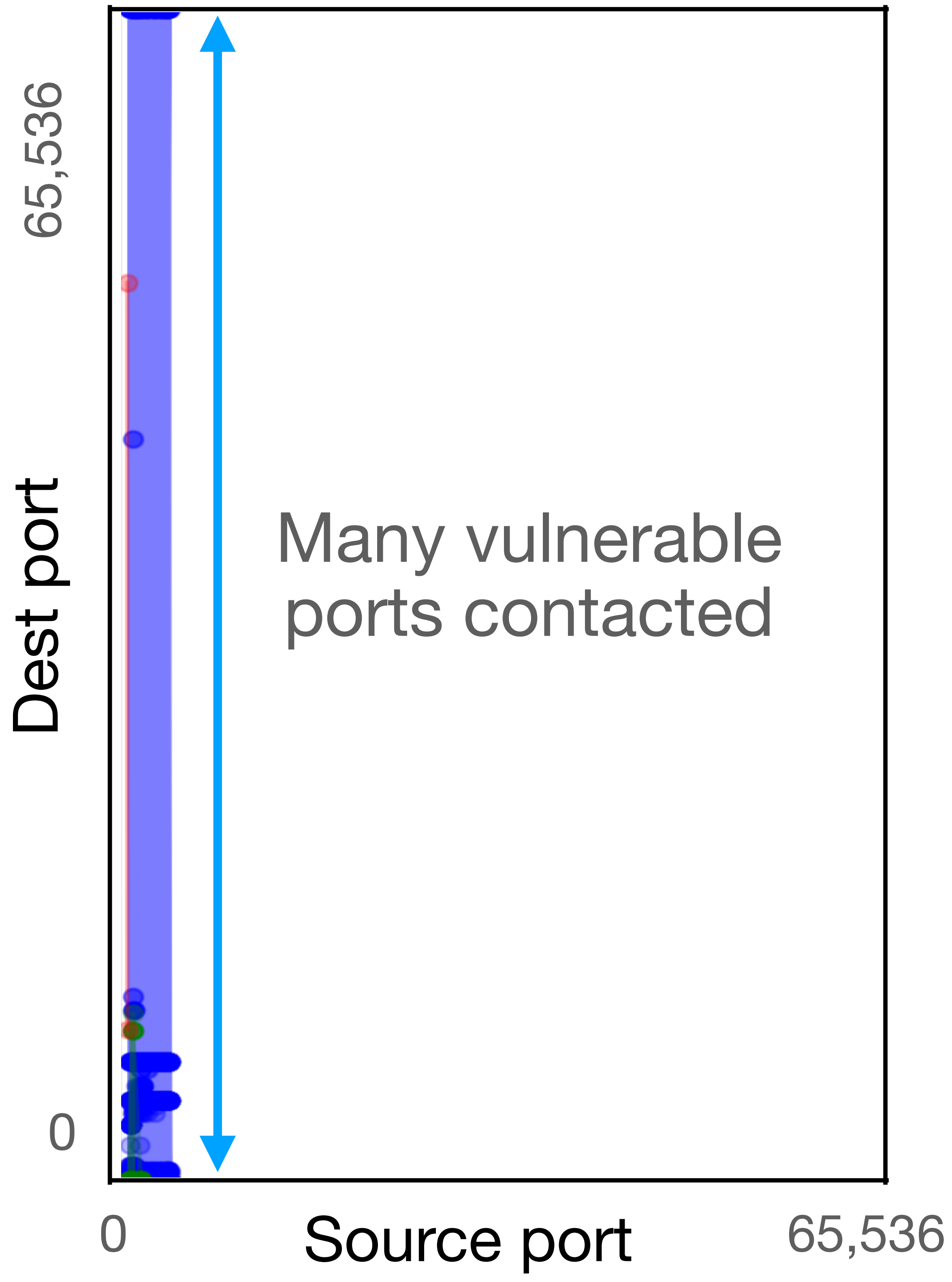
First observations on CTU-13

Inspecting the communications of two different hosts

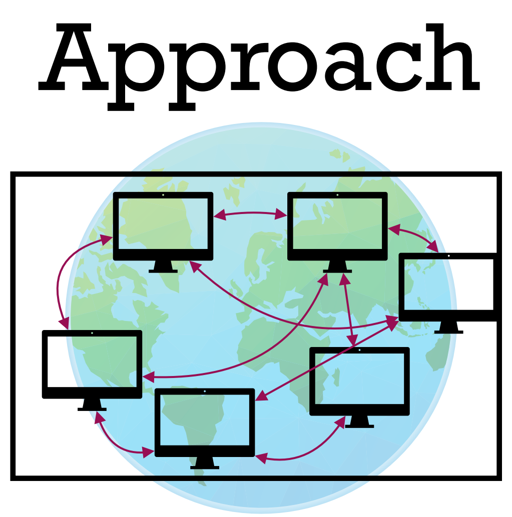
Benign host



Infected host (bot)



Frequency distribution of protocol uses



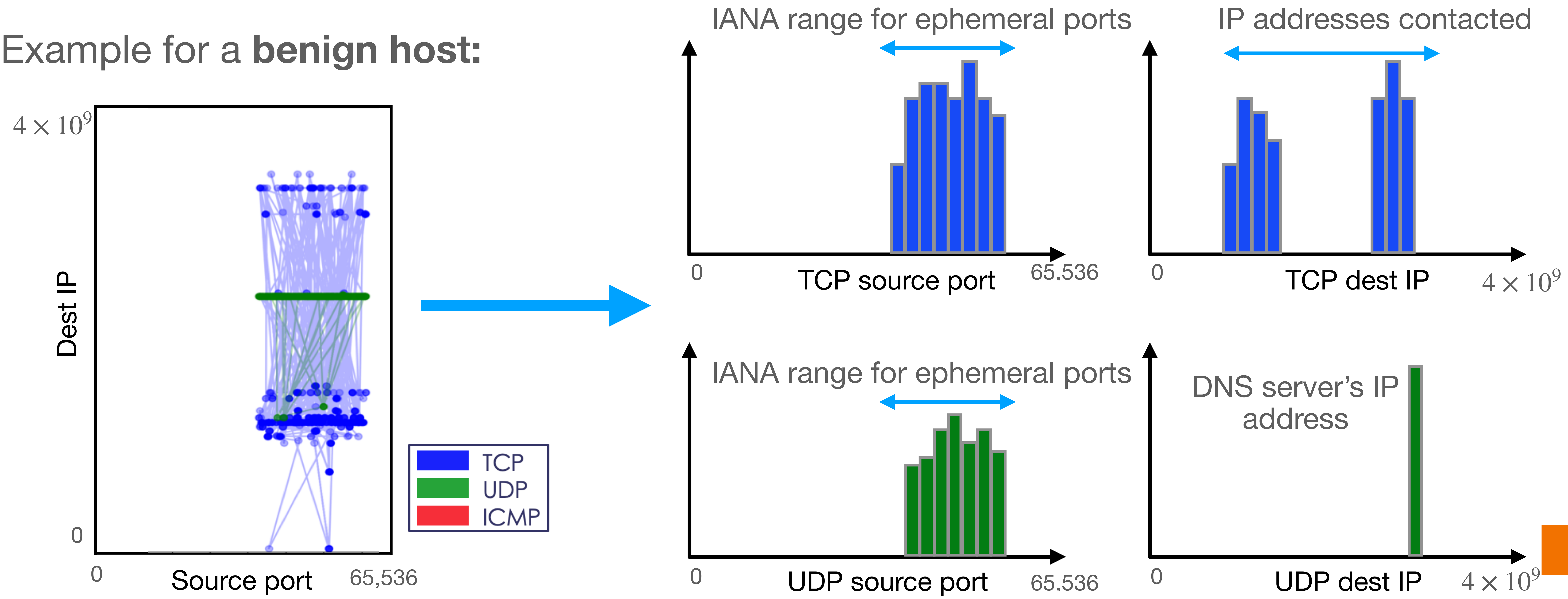
Host signature: **concatenation** of the **frequency distributions** of the 9 features:

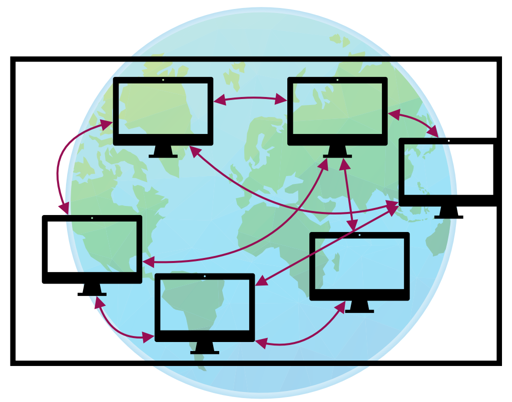
9 features from the combination of:

- TCP
- UDP
- ICMP

- Source port
- Destination port
- Destination IP address

Example for a **benign host:**





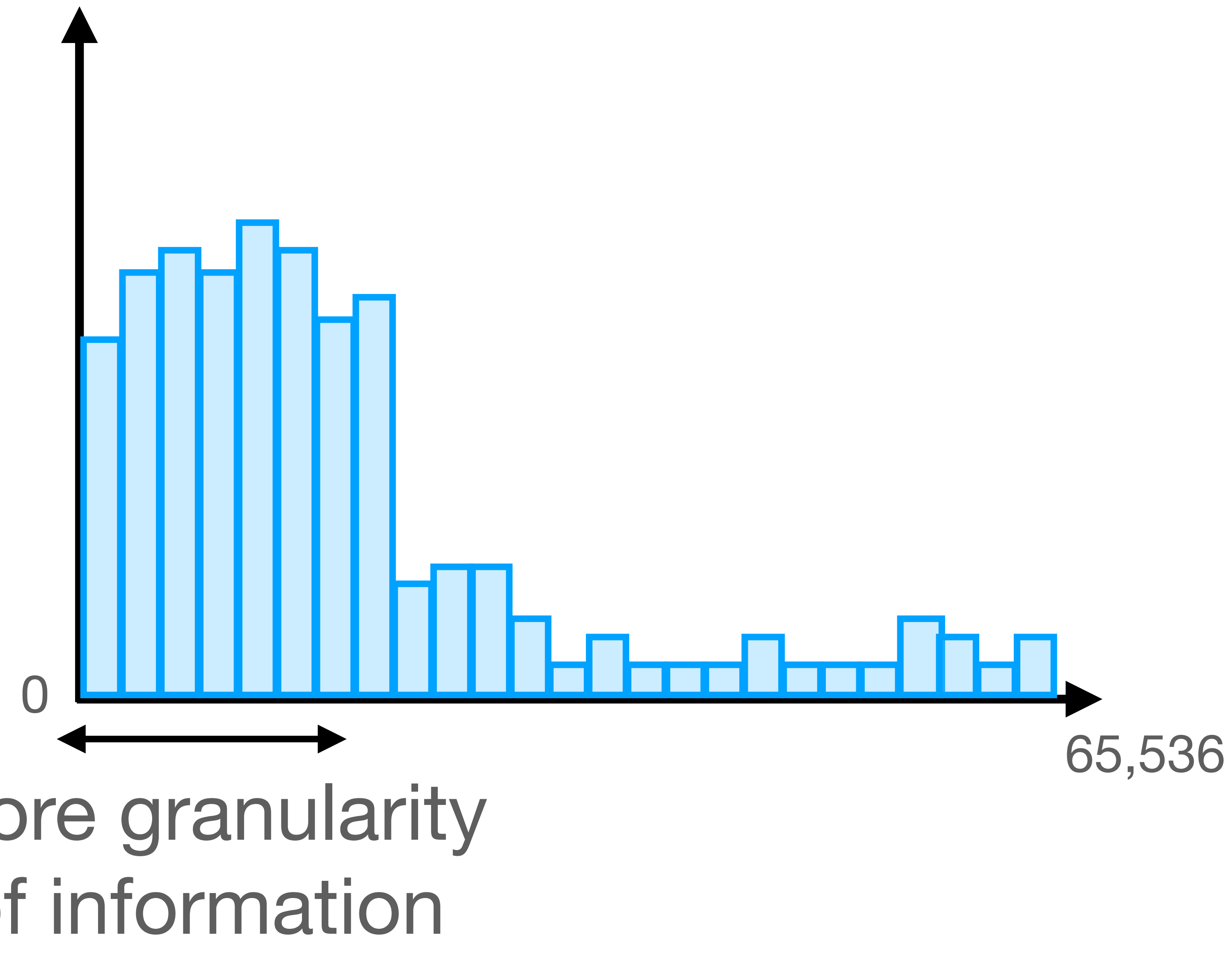
Quantisation bin

Adaptive bin width computed for each attribute: the same bin distribution for all hosts

Example for a **Destination port TCP:**

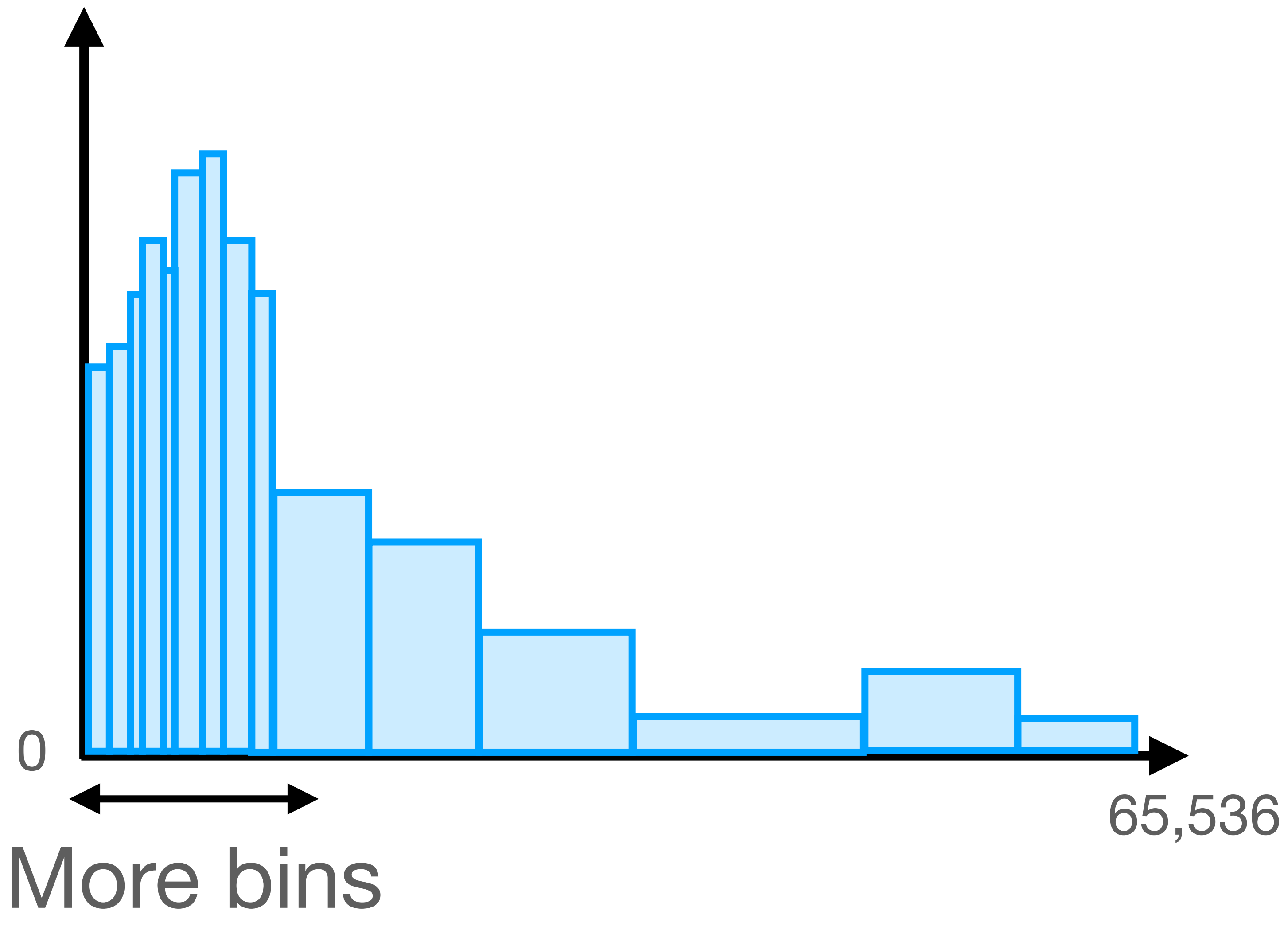
Regular bins

Bins of equal width



Adaptive bins

Bins width adapted to the density of information



Our general approach: BotFingerPrinting

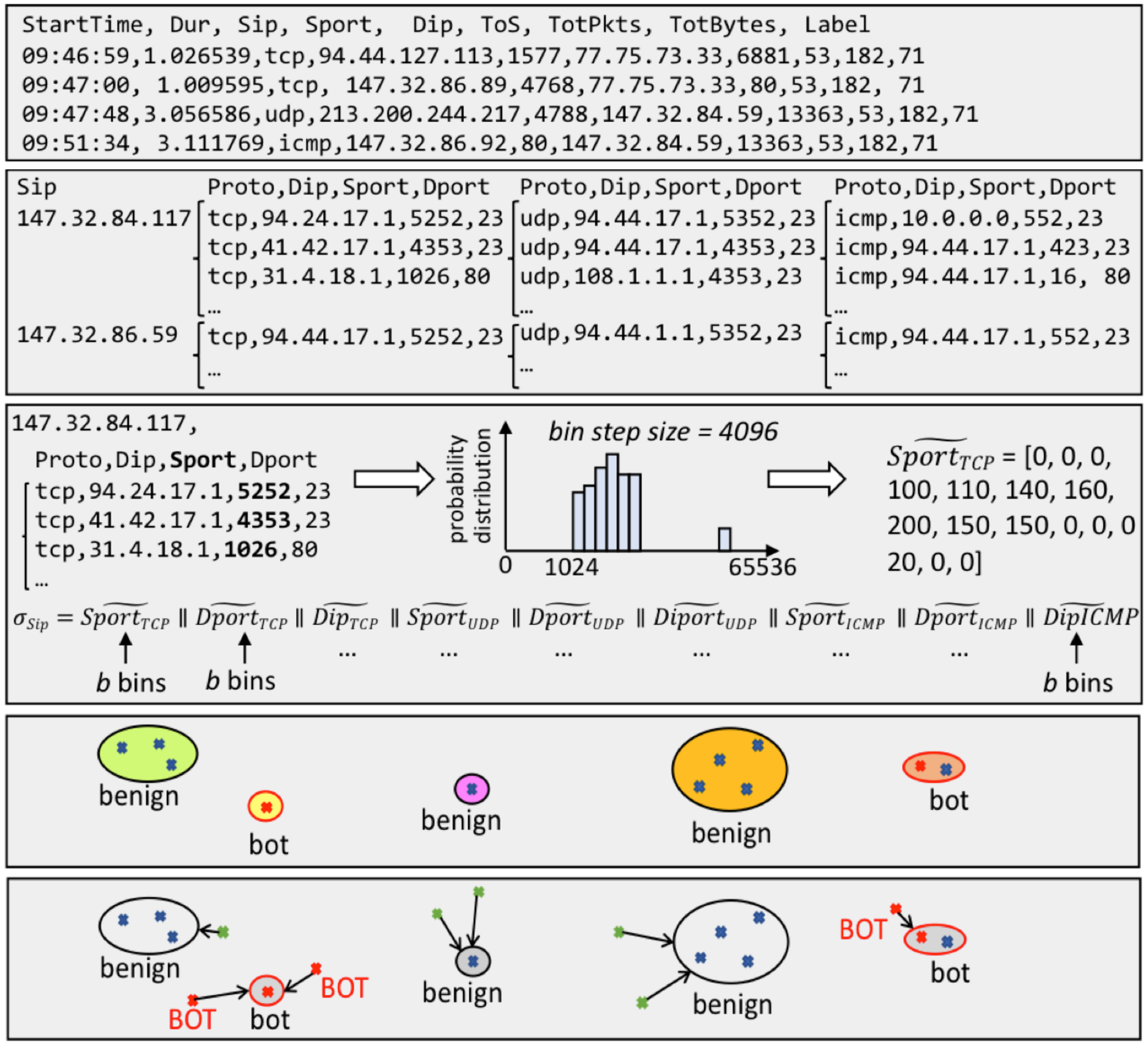
1 **Flow records collection**

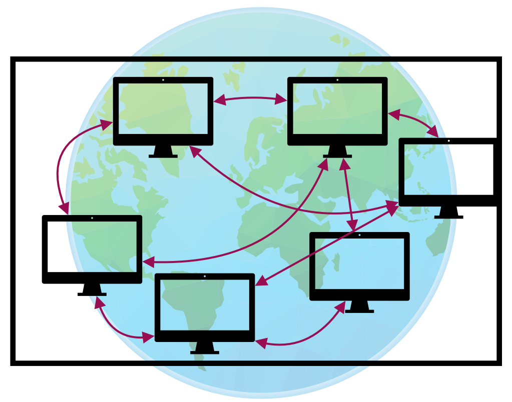
2 **Host network Sip filtering and grouping**

3 **Quantification** (frequency distributions computation)

4 **Offline training** (clustering)

5 **Online classification** (distances computation)





Evaluation

- Tuning depending on the goal(s) to favour
  - Maximising the true bot detection
  - Minimising the false positive rate
  - Minimising the memory usage

"An empirical comparison of botnet detection methods," Computers & Security, 2014.

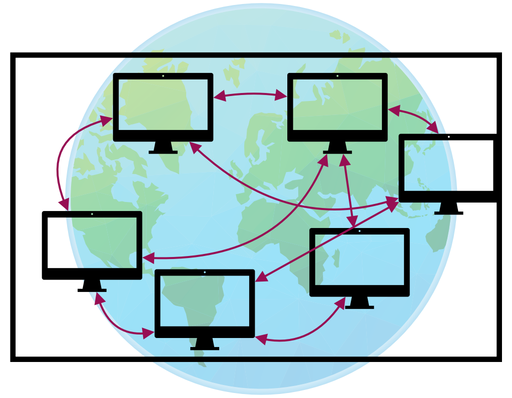
"BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation," Usenix Security Symposium, 2007.

"BotGM: Unsupervised Graph Mining to Detect Botnets in Traffic Flows," CSNet, 2019.

"A Graph-Based Machine Learning Approach for Bot Detection," IFIP/IEEE, 2019.

Accuracy of state-of-the-art techniques and BotFP

→ BotFP accuracy **between 97 and 100%**



BotFP conclusion

Benefits of the detection inspecting **service usages:**

- **Histograms approximate the relations between hosts**
- Far **more lightweight** and **more efficient** than graph-based approaches
  - Very high accuracy (from 97 to 100%), outperforming other state-of-the-art techniques
  - Nearly all bots detected with very few false positives

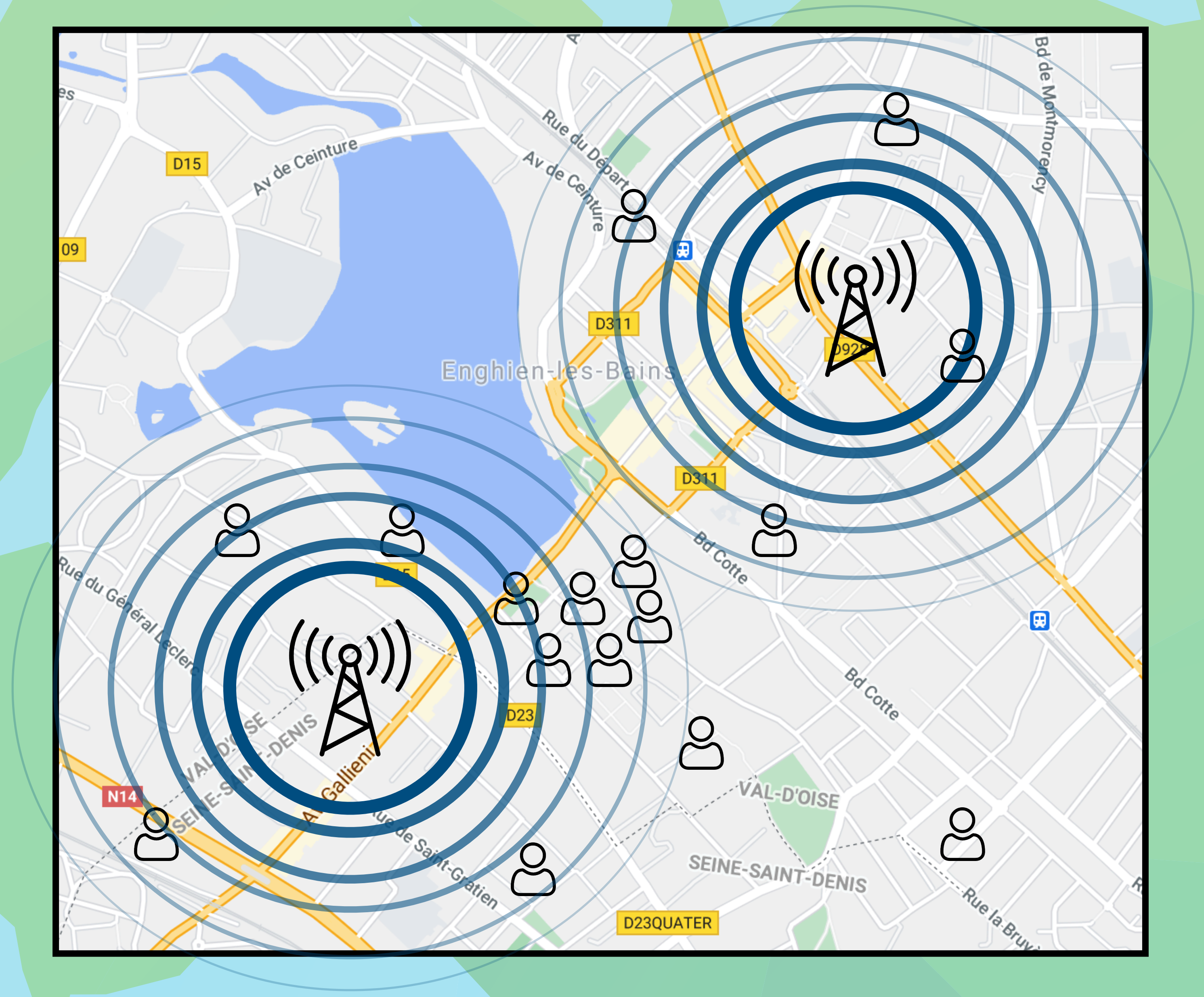
ASTECH

ASTECH

In cellular networks

ASTECH

In cellular networks



- Detecting **uncommon behaviours from users**
- **Special events** and outages from the operator
- Impacted mobile **applications** during the event

ASTECH

Challenge:

- Detection of events in mobile traffic not tackled at the **app-level**
- **Classification** of events **rarely done**

Our approach:

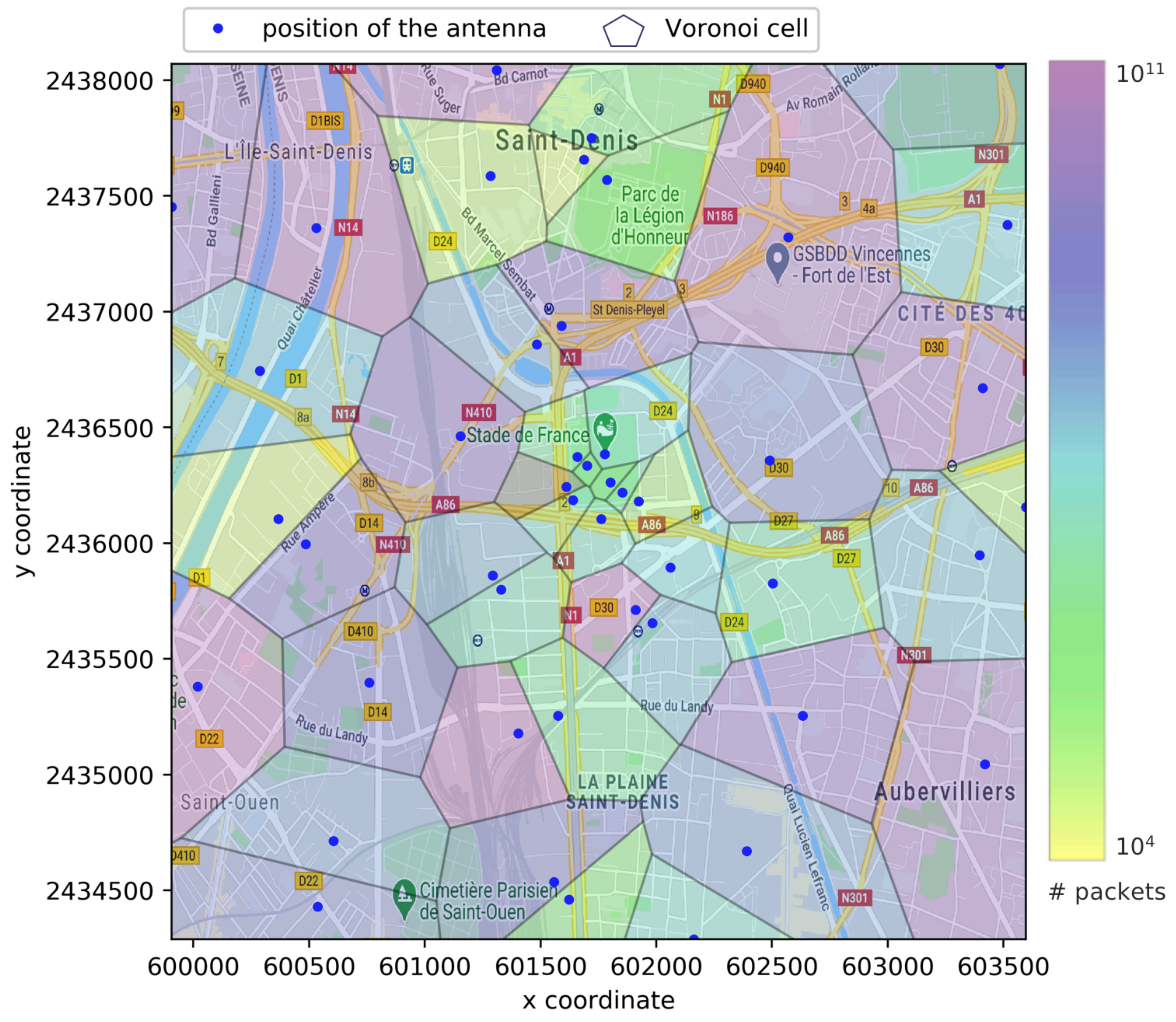
- Spatiotemporal convex hull anomaly detection
- Analysis of **impacted mobile apps** and spatiotemporal spreading

Our contribution: detection and **classification** of events in mobile traffic

Evaluation on citywide traffic data

TCP sessions aggregated by **time series** per:

- **Mobile application**
- **Attribute:** #users, upload and download traffic
- **30-minute time slot**
- **Base station**

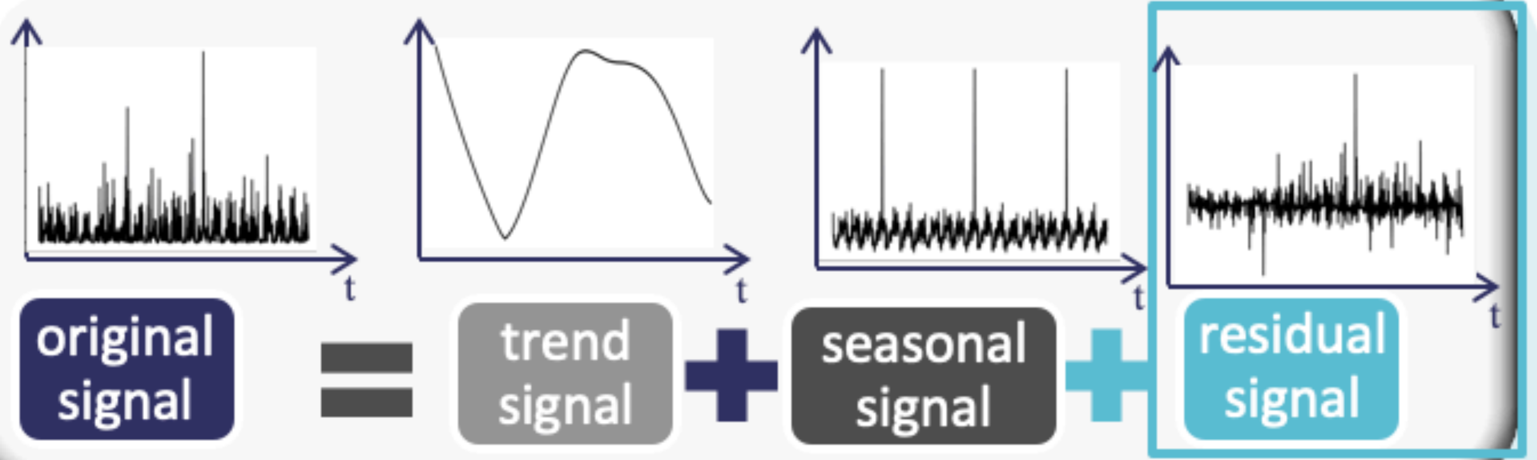


ASTECH

Step 1: set of time series $\mathcal{Y}$

For each time series (i.e. each app, feature and base station):

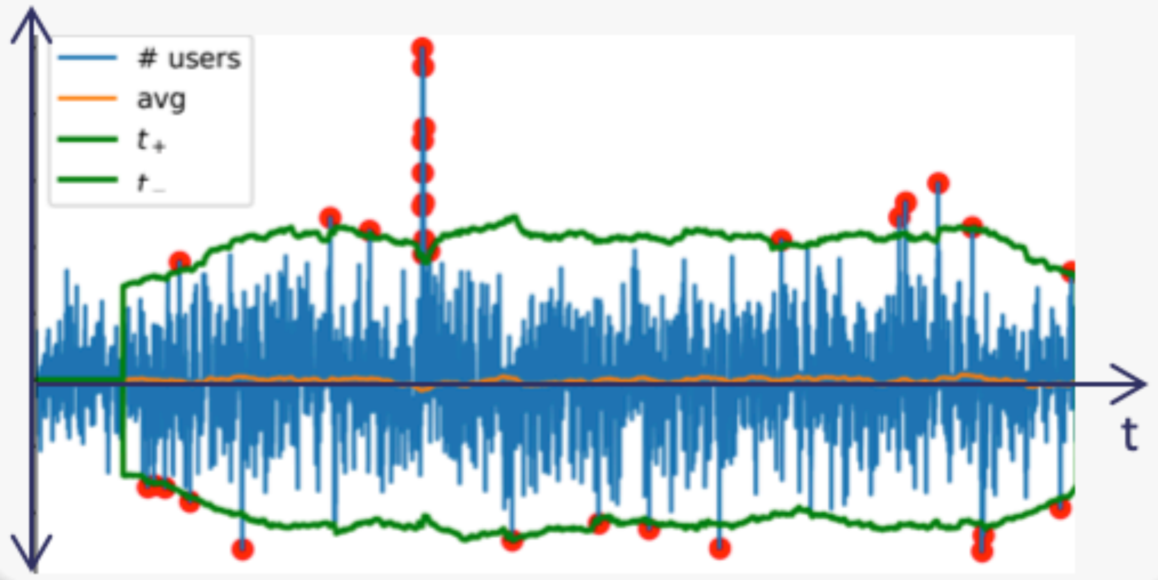
Time series decomposition



extraction

Time series anomaly detection

- **Smoothed Z-score:**
  $t_{\pm} = \boldsymbol{avg} \pm \tau * \boldsymbol{std}$ ($\tau = 3.5$) with $\boldsymbol{avg}$ and $\boldsymbol{std}$ the mean and standard deviation on period $[t_0\text{-lag};t_0]$
- Anomaly if value(t_0) > t_+ or < t_-



Step 2.1: set of raw anomalies $\mathcal{A}$

Given the whole set of raw anomalies:

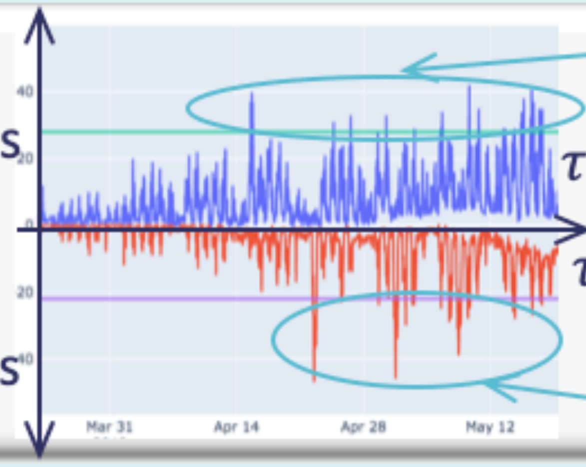
Snapshots formation

Snapshot ⇔ group of anomalies at a given place and time
Gives an idea about the current network's state

Step 2.2: set of snapshots $\mathcal{S}$

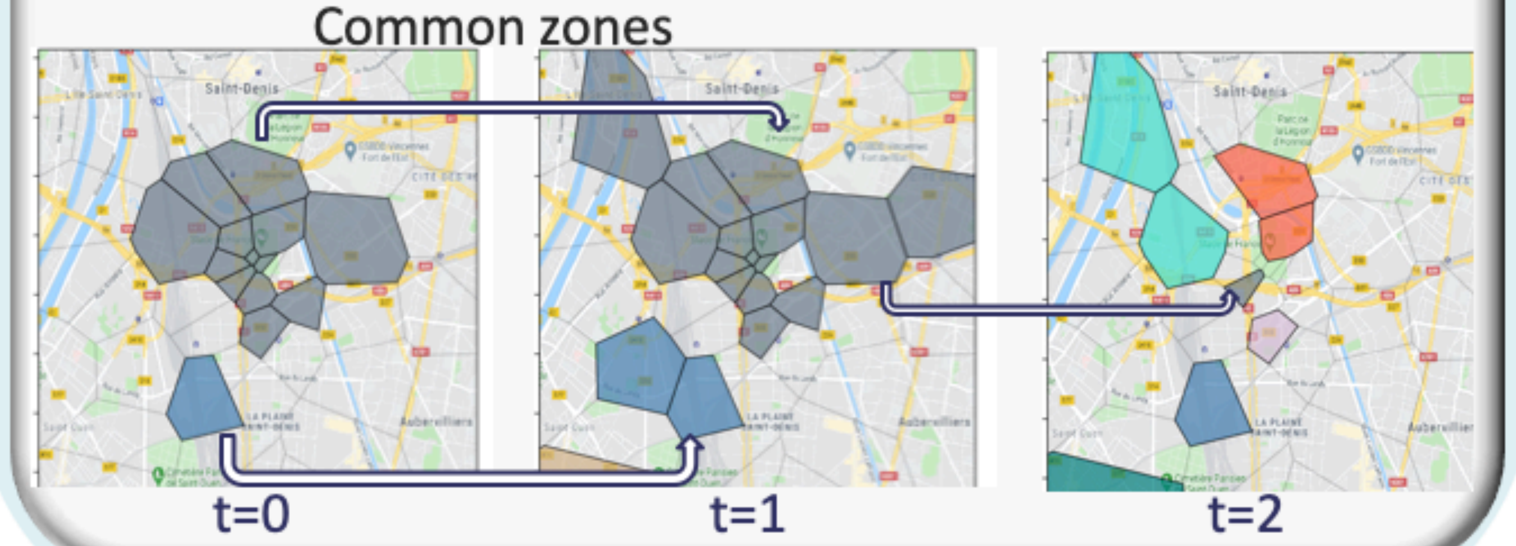
Selection of abnormal snapshots

For each zone z:



2-step clustering

- **Recursive algorithm by region growing**
- **+ Aggregation into spatio-temporal events**

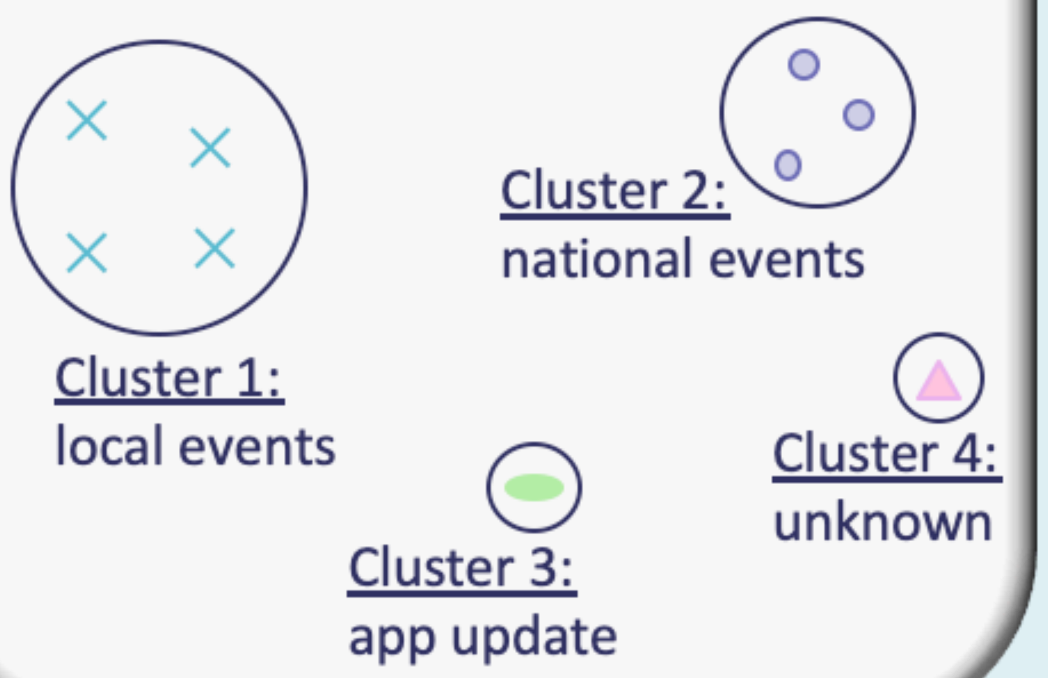


Step 3: set of group anomalies Γ

Given the whole set of group anomalies:

Event classification

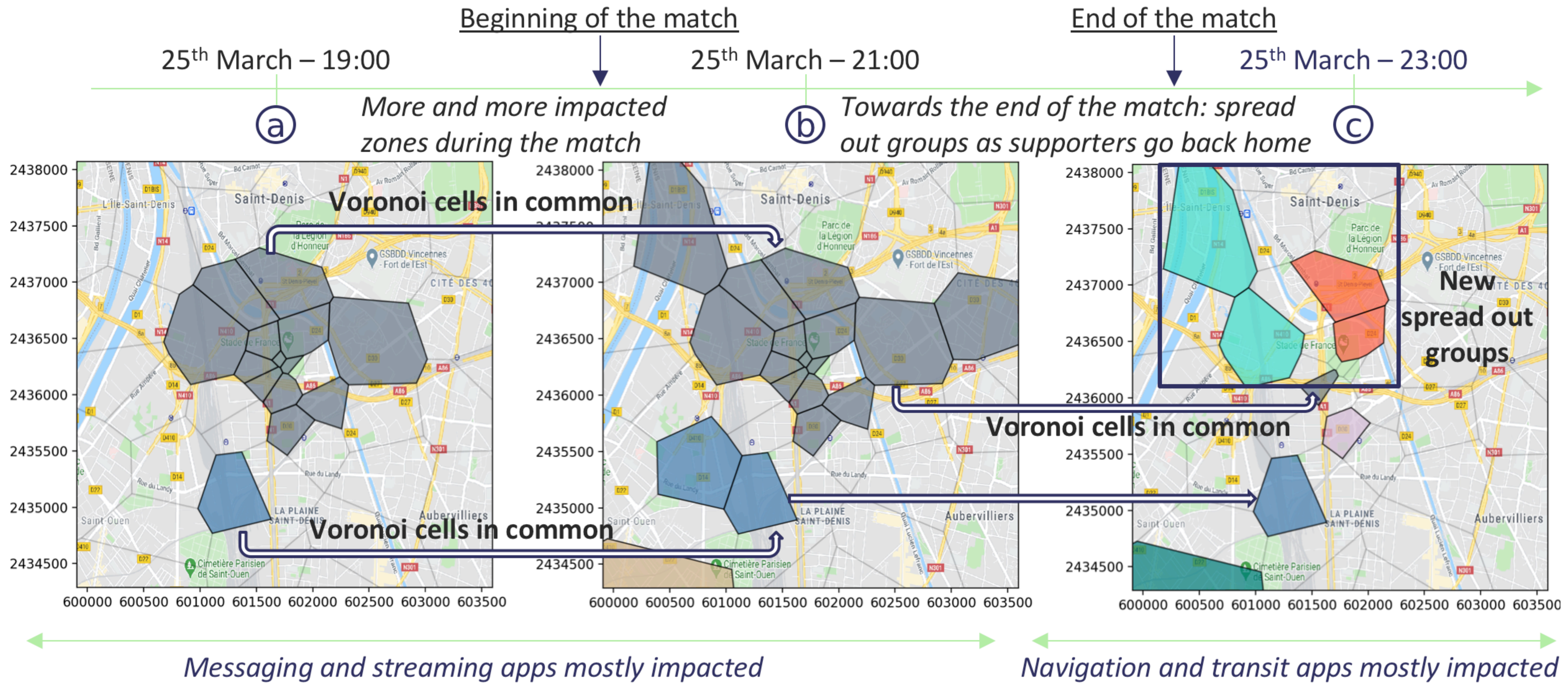
- **3 super-features to classify group anomalies:**
  - Sign of traffic variations
    i.e., less or more traffic than usual
  - Anomaly sparsity
    Centralized or distributed anomaly?
  - Group of impacted apps
    Rather a single or several impacted apps?
- ***k*-means clustering of anomalies Γ based on their super-features**



Formation of <u>spatiotemporal groups</u>

1. **<u>Spatial grouping</u>**: abnormal snapshots into spatial groups

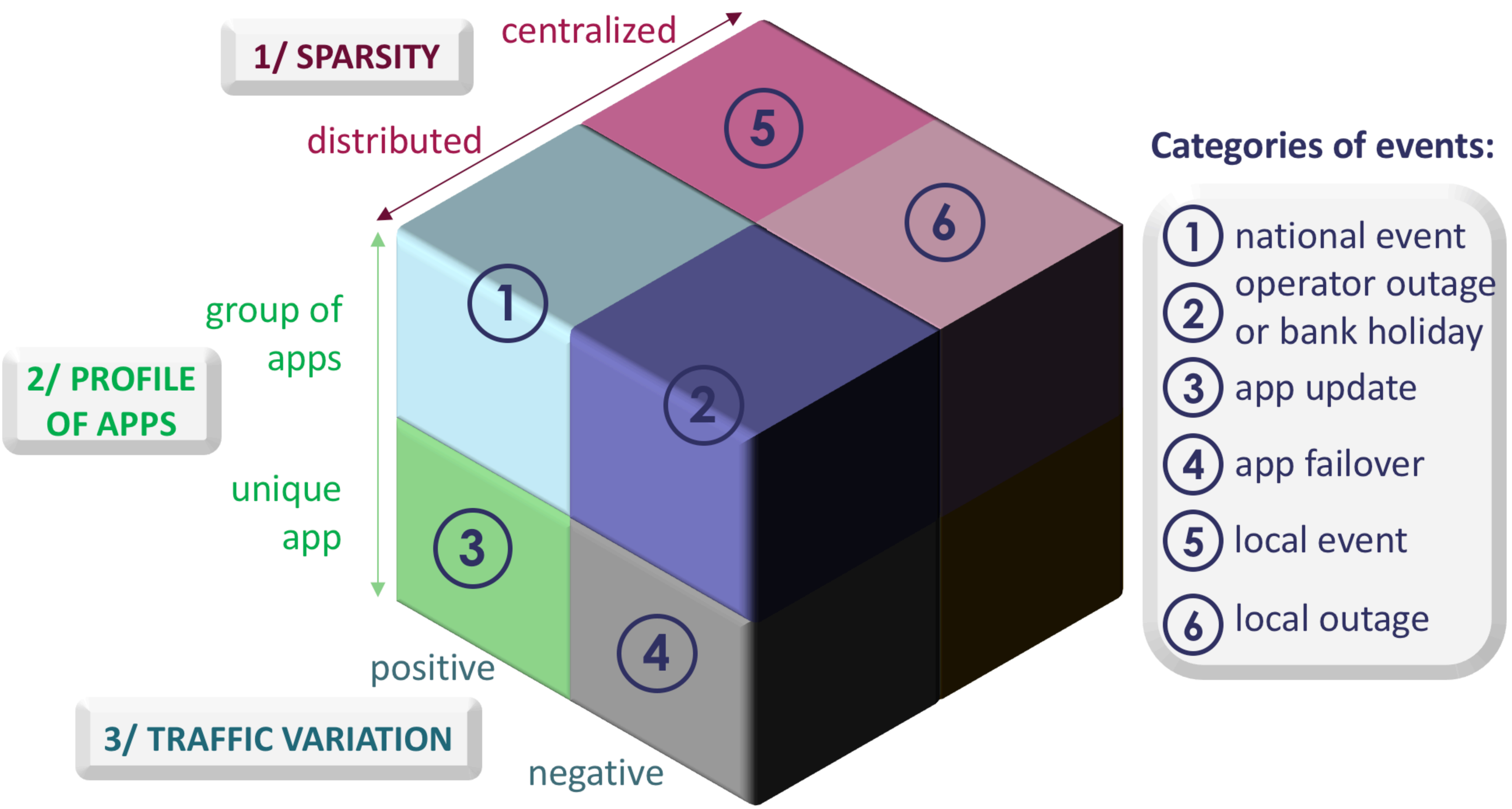
2. **<u>Spatiotemporal grouping</u>**: spatial groups into spatiotemporal **group anomalies**



Special events characterisation

Clustering to group similar events into broad categories

- 3 super-features
- 8 broad categories of events



Evaluation on citywide dataset

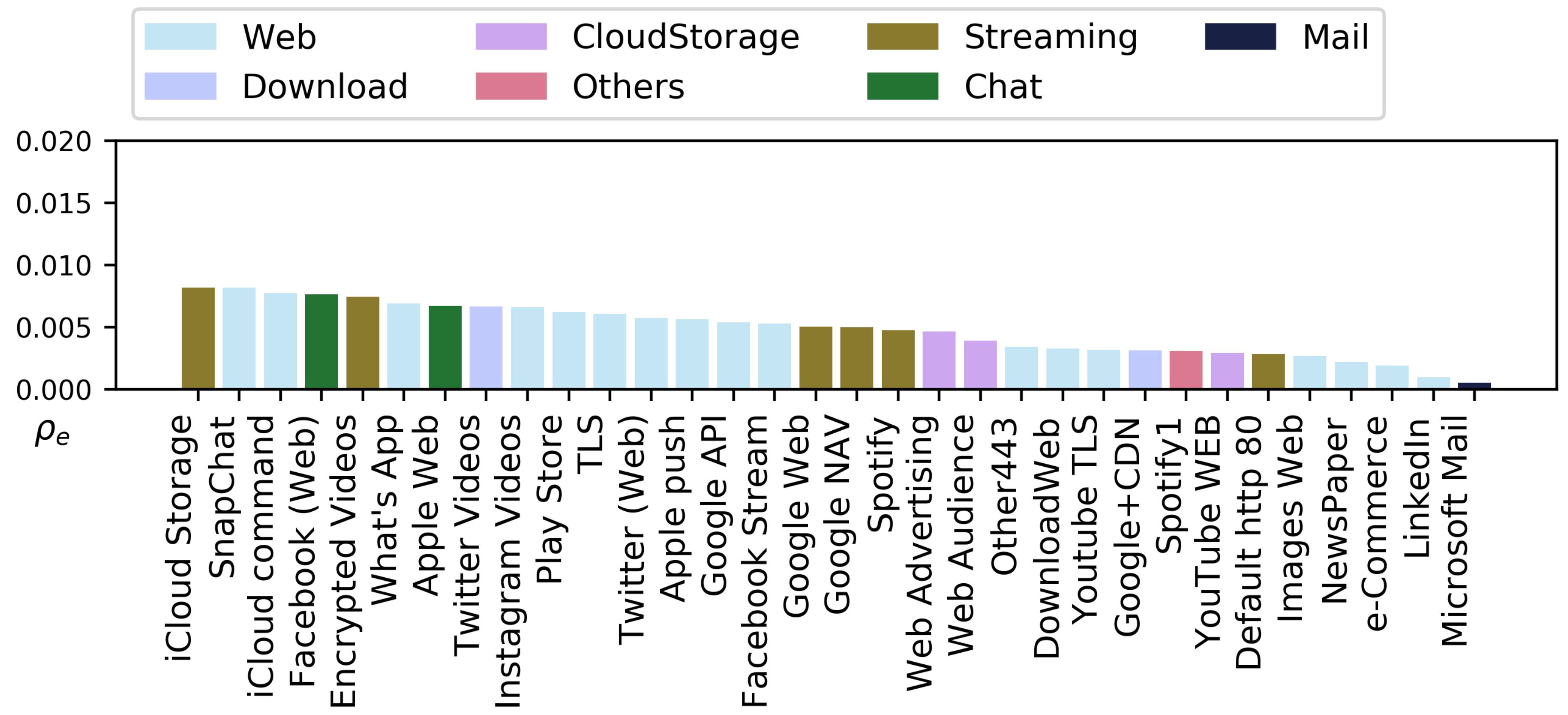
- Events of **positive** anomalies
  - Matches/concerts at Stade de France
  - Notre-Dame de Paris fire
  - Application update
- Events of **negative** anomalies
  - Bank holidays
  - Orange 4G network outage
  - Google Cloud outage

Evaluation

Typology of impacted apps

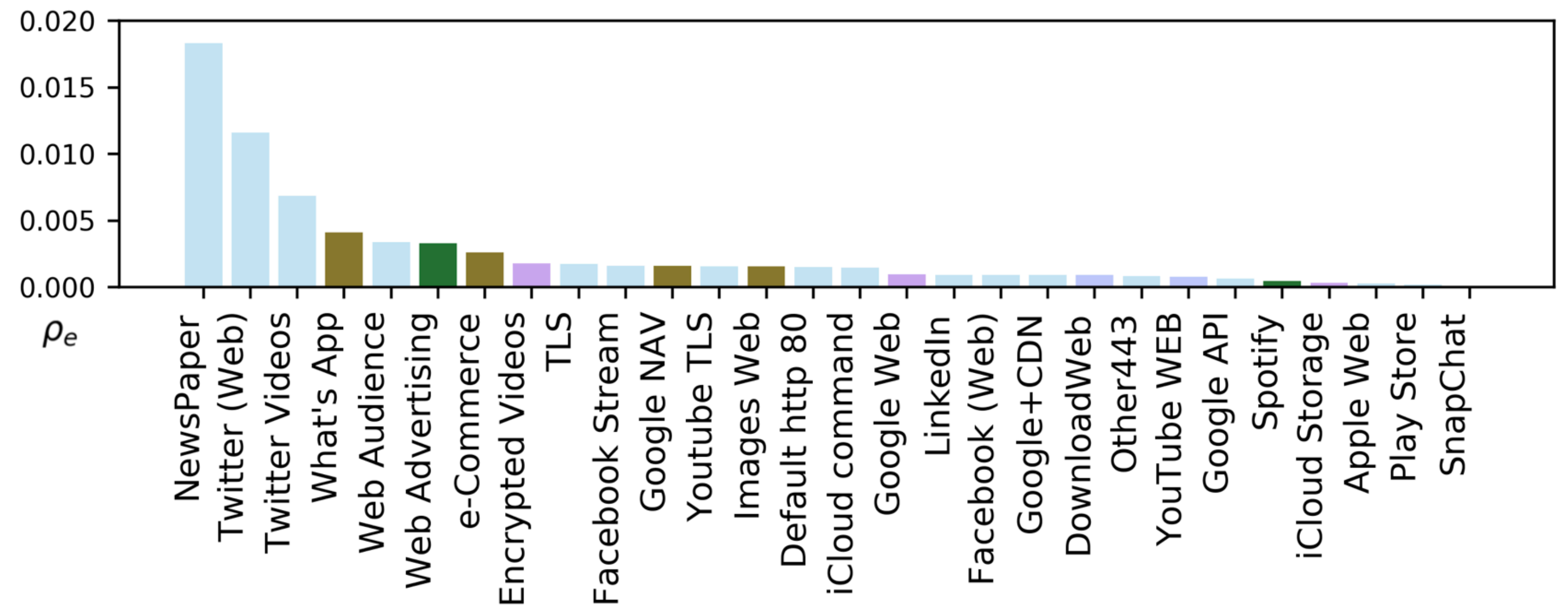
Events of <u>positive</u> anomalies

Local event



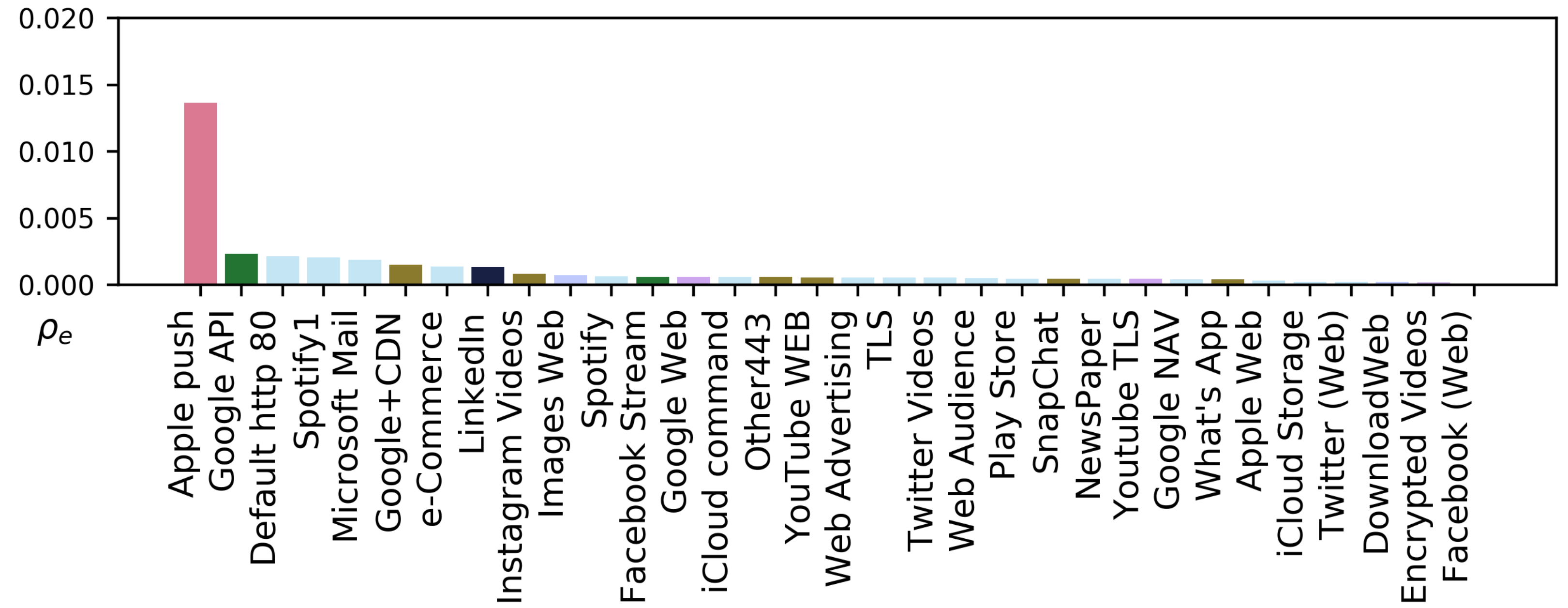
→ <u>Several apps:</u> streaming, web and chat applications

National event



→ <u>Several apps:</u> NewsPaper and Twitter

App update



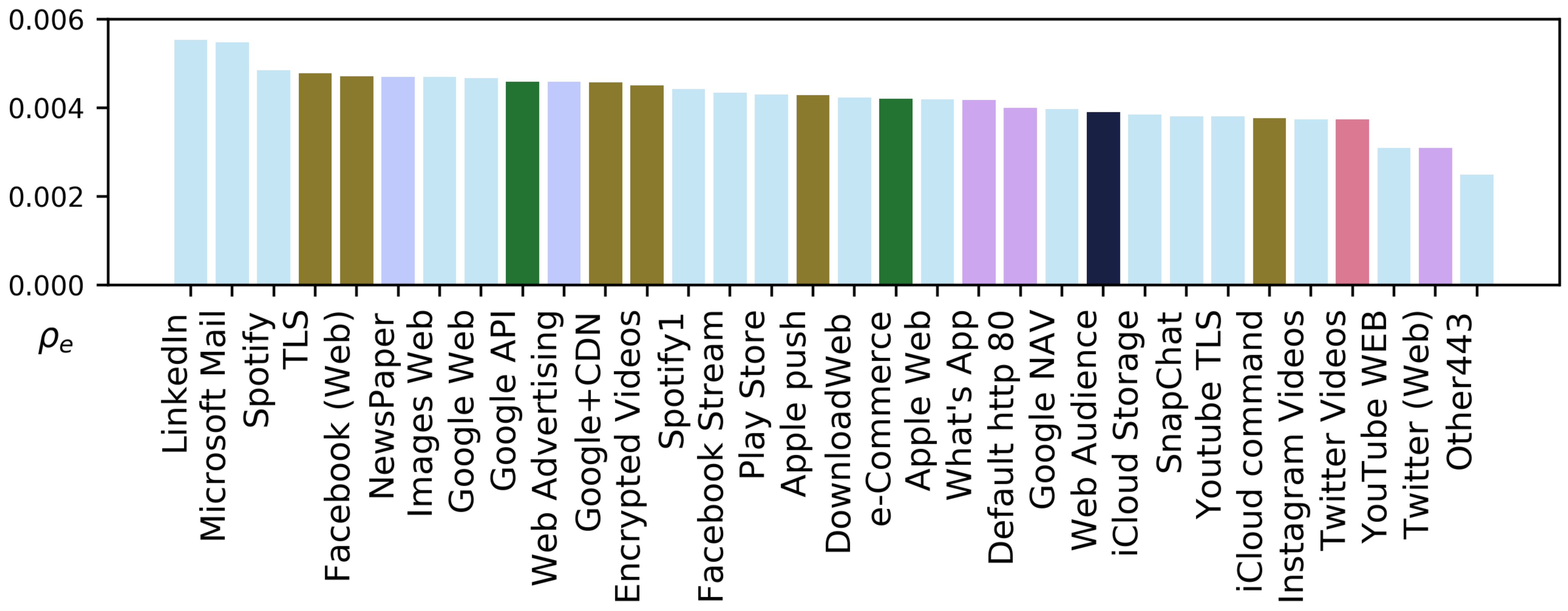
→ <u>Unique app:</u> Apple push

→ **Specific typology** for the events of **positive** anomalies

Typology of impacted apps

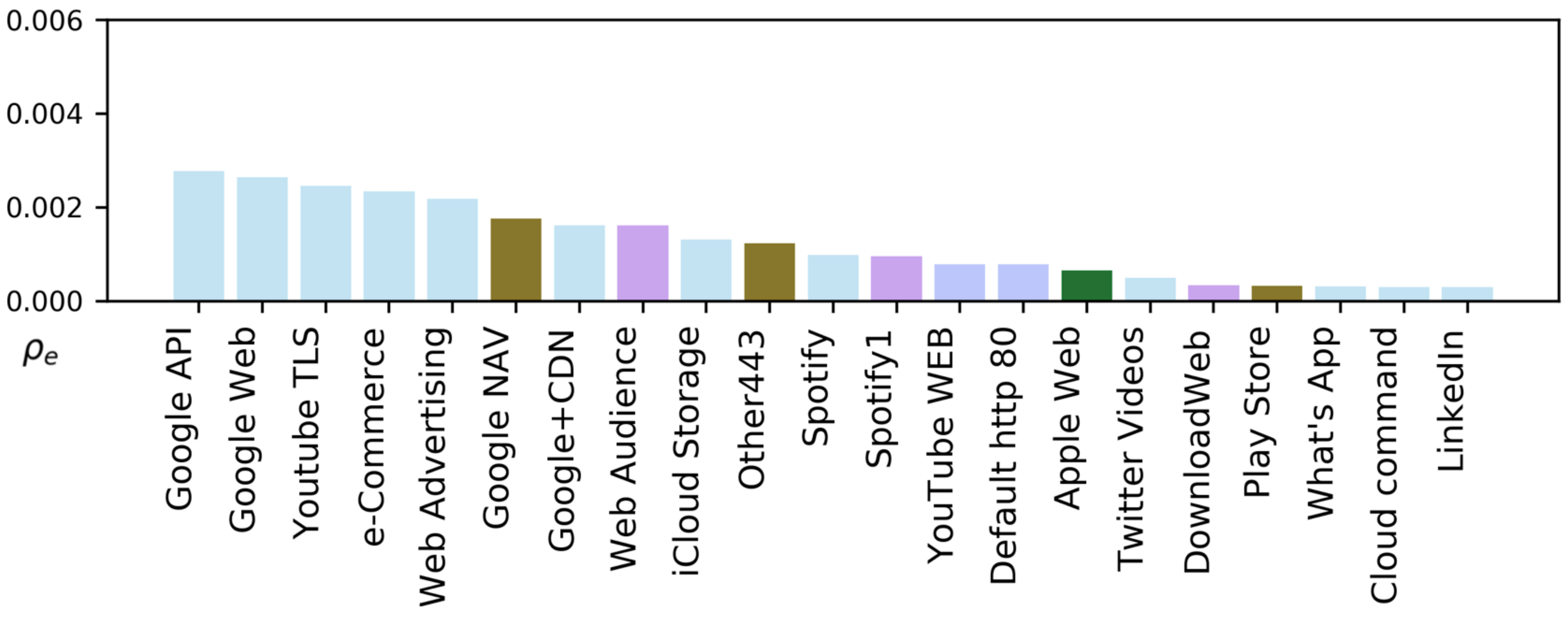
Events of negative anomalies

Bank holiday



→ **All apps impacted**

Outage



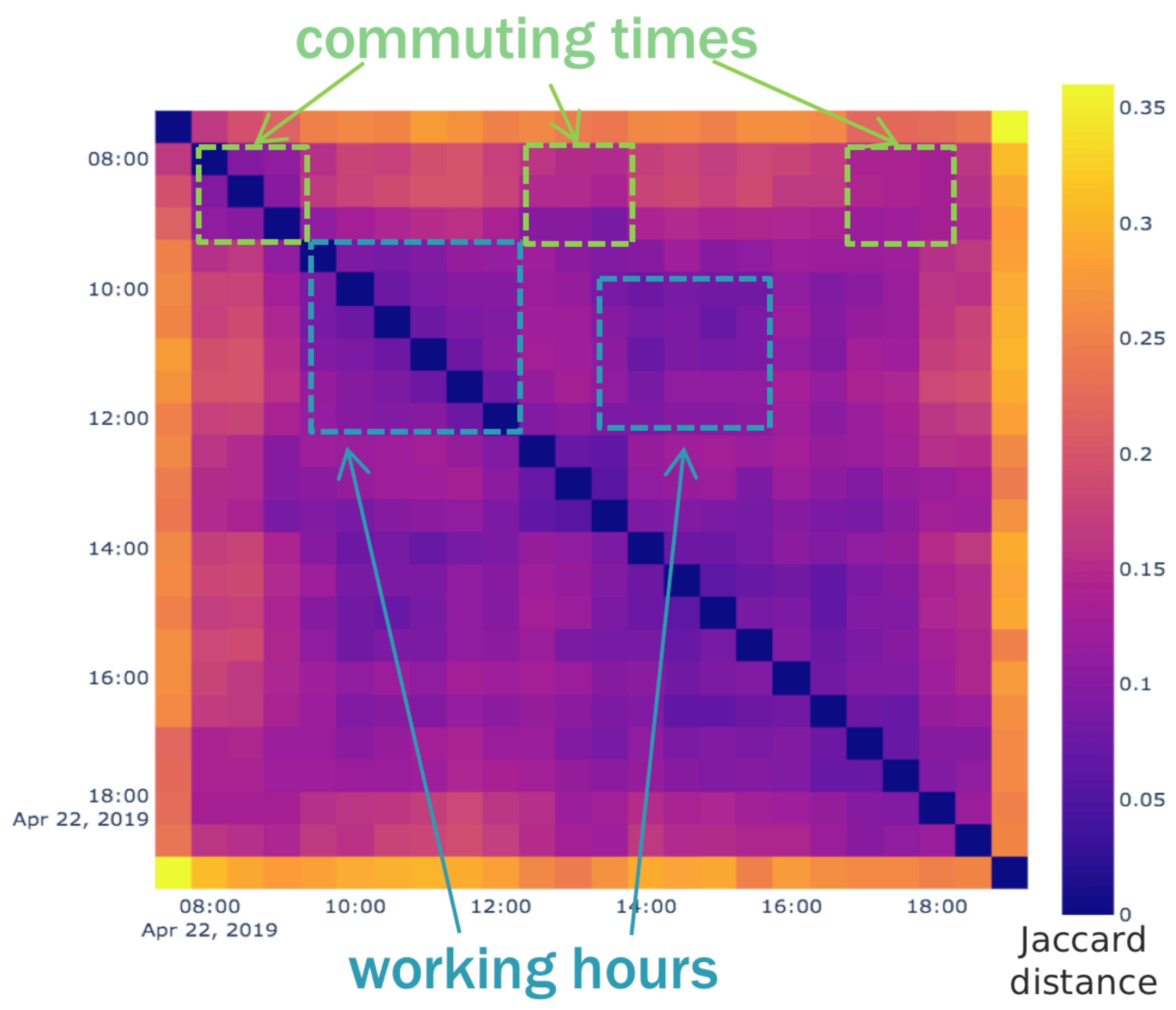
→ **All apps impacted**

→ **No specific typology** for **negative** anomalies, as **all apps** (lightly) impacted

Temporal evolution of the set of impacted apps

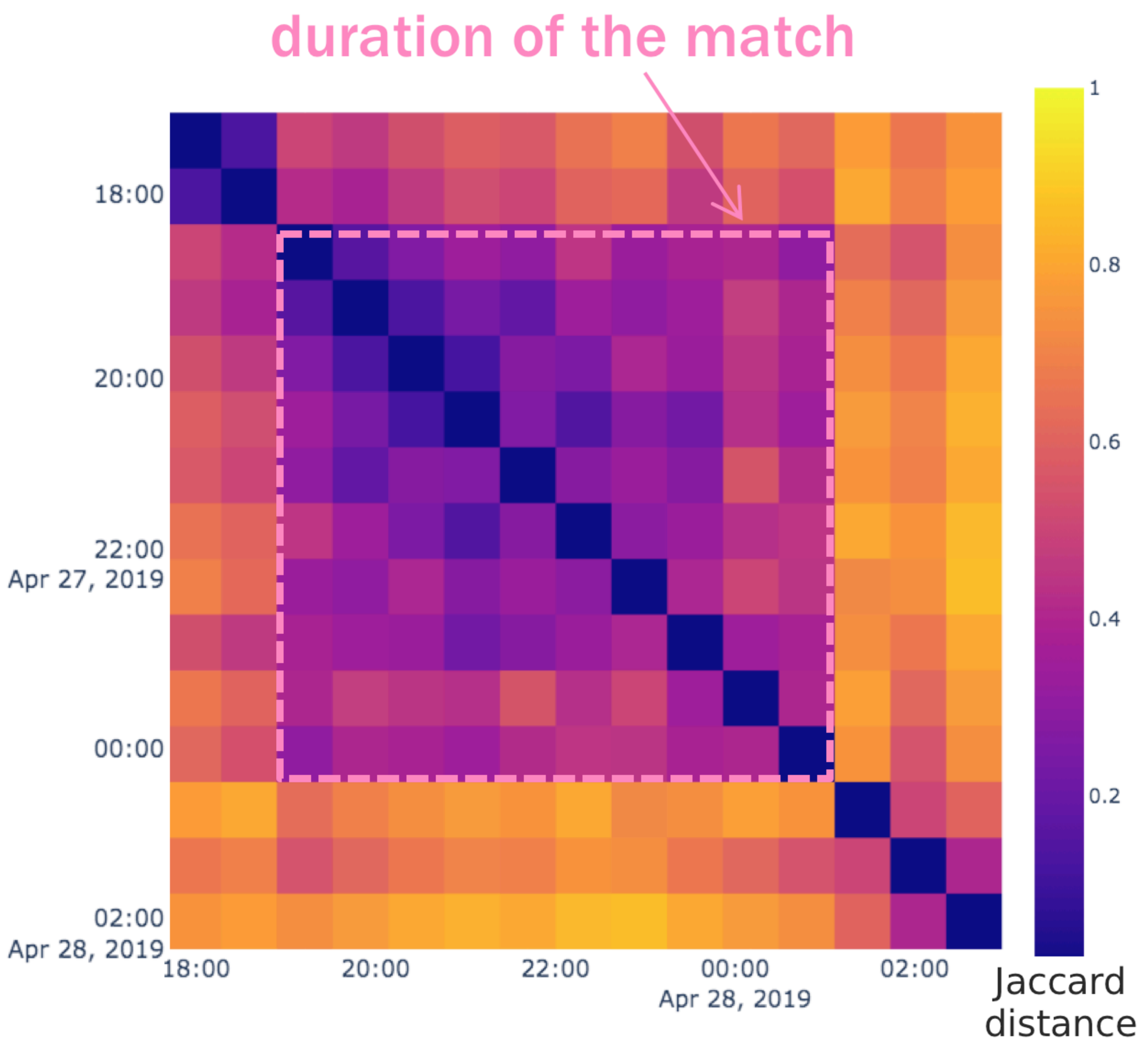
Bank holiday

- Similar profile (darkest squares) during commuting times
- Similar profile during working hours



Outage

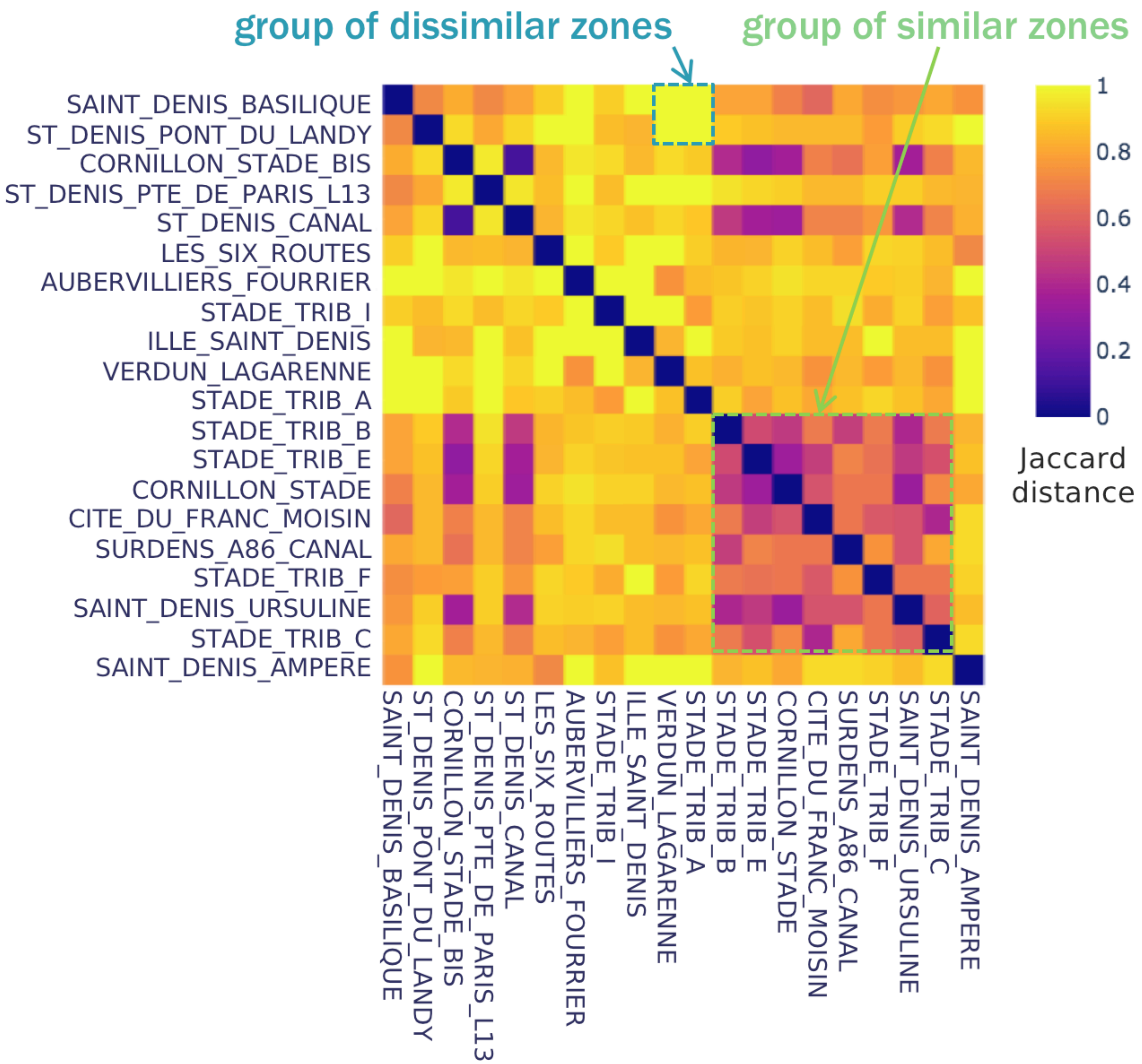
- Similar profile at the heart of the match



Spatial evolution of the set of impacted apps

One generic pattern: bank holidays and local/national events

- Very close Voronoi cells while others more distant



ASTECH conclusion

Benefits of studying the mobile apps usage:

- **Spatiotemporal** group anomaly detection
  - Fine characterisation of a wide variety of events

- **Typology of impacted applications**
  - Events of **positive** anomalies → **typology** of impacted applications (either subset of apps or unique app)
  - Events of **negative** anomalies → **no specific** typology

General conclusion

Contributions

Split-and-Merge

- Early detection of **emerging botnets** spreading on the Internet

- A. Blaise, M. Bouet, S. Secci, V. Conan, "Split-and-Merge: Detecting Unknown Botnets," *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2019.
- A. Blaise, M. Bouet, V. Conan, S. Secci, "Detection of zero-day attacks: An unsupervised port-based approach," in *Elsevier Computer Networks*, vol. 180, pp. 107391, 2020.
- A. Blaise, S. Scott-Hayward, S. Secci, "Scalable and Collaborative Intrusion Detection and Prevention Systems Based on SDN and NFV," chapter in *Guide to Disaster-Resilient Communication Networks, Computer Communications and Networks,* pp. 653-673, Springer, 2020.

Contributions

BotFingerPrinting

- Detecting **botnet infected hosts** at the enterprise-level
- Histogram-based algorithm to model communications

- A. Blaise, M. Bouet, V. Conan, S. Secci, "Botnet Fingerprinting: A Frequency Distributions Scheme for Lightweight Bot Detection," in *IEEE Transactions on Network and Service Management*, vol. 17 (3), pp. 1701-1714, 2020.
- A. Blaise, M. Bouet, V. Conan, S. Secci, "BotFP: FingerPrints Clustering for Bot Detection," *IEEE/IFIP Network Operations and Management Symposium (NOMS), 2020.*

Contributions

ASTECH

- Detection of spatiotemporal events occurring in a city, in terms of volume and impacted apps

- A. Blaise, M. Bouet, V. Conan, S. Secci, "Group anomaly detection in mobile app usages: a spatiotemporal convex hull methodology," *submitted to IEEE Transactions on Mobile Computing.*

General perspectives

- Demonstration of the **potential** of the analysis of **port numbers, mobile applications and services**
  - Act as universal (in all subnetworks) and permanent identifiers
  - **Efficient** and **lightweight** algorithms
- **Real time** implementation: **online** algorithms
- **System applicability**
- Development of **hybrid** solutions: coupling the analysis on flows and IP addresses with port numbers

Perspectives

- **Split-and-Merge**
  - Implementation in a **Software-Defined Networking** environment
  - P4 network programming language: detection, attack mitigation

- **BotFingerPrinting**
  - Exploring unsupervised learning techniques
  - Real time implementation in a **Security and Information Event Management**

- **ASTECH**
  - Grouping anomalies disconnected from each other
  - Real time implementation in **5G Platform**

Thank you